HARNESSING HETEROGENEOUS ASSOCIATION IN REAL-WORLD NETWORKS

BY

YU SHI

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Doctoral Committee:

Professor Jiawei Han, Chair
Professor Hari Sundaram
Professor Jian Peng
Doctor Myunghwan Kim, LinkedIn Corporation

ABSTRACT

Real-world networks often contain heterogeneity due to the heterogeneous nature of the world. A few examples of such networks include multi-view social networks, heterogeneous bibliographic networks, biomedical networks, *etc.* Ostensibly the heterogeneity of real-world network appears as the typed essence of nodes and edges. By considering type information, researchers have shown that using the typed networks can achieve performance better than using the homogeneous networks in a wide variety of downstream applications such as classification, clustering, recommendation, and outlier detection.

Beyond the low-level heterogeneity in nodes and edges on the surface, their types also naturally induce higher-level typed network components. In my practice mining real-world networks, I identify that the heterogeneity also prevalently lies in the association across different network components, and such heterogeneous association is often important and intrinsic to the information embodied in the networks.

In this dissertation, I investigate the necessity of modeling heterogeneous association in real-world networks and develop methodologies to simultaneously leverage the rich information and accommodate the incompatibility in the presence of heterogeneous association. A series of new models along this line are proposed for specific problems including learning network embedding, defining relevance measures, and discovering hypernymy relation, together with the discussion on how the principles reflected by these models can be used in other network mining tasks. These proposed models cannot only achieve better quantitative results but also uncover the semantics hidden in the heterogeneous association of real-world data.

To my family for their love and support.

ACKNOWLEDGMENTS

I would like to thank all the people who generously provided me with tremendous support through my Ph.D. study and made this dissertation possible.

Firstly, I wanted to thank Professor Jiawei Han for his guidance and support along the way. Without a degree in computer science prior to the Ph.D. study, I had an uneasy start in my data mining research, but he guided me through the difficulties with great patience. From him, I have learned so many lessons about identifying important problems, proposing effective solutions, and extracting meaningful insights. Moreover, he has also been a role model by showing me how to collaborate with others, present ideas, lead a team, take initiatives, and deal with setbacks. Being fortunate to be advised by him has been my life-changing experience and will always be a treasure in my life.

I would like to thank my committee members, Professor Hari Sundaram, Professor Peng Jian, and Doctor Myunghwan Kim. Not only they have provided valuable advice for this dissertation, but for years, I have also enjoyed and benefited from their enlightening courses or the caring and meticulous internship mentoring.

I have also been fortunate enough to be offered direct, hands-on advice from multiple senior group members in DMG at UIUC, Honglei Zhuang, Huan Gui, Fangbo Tao, and Chao Zhang. Many thanks to the other collaborators and friends in the extended DMG family. Alphabetically, they are Po-Wei Chan, Joe Chen, Xiusi Chen, Yucheng Chen, Bolin Ding, Ahmed El-Kishky, Xiaotao Gu, Fang Guo, Wenqi He, Xinwei He, Jiaxin Huang, Meng Jiang, Dongming Lei, Min Li, Jessie Zhenhui Li, Ji Li, Qi Li, Yuchen Li, Zoey Li, De Liao, Liyuan Liu, Jialu Liu, Zhengzhi Lou, Stephen Macke, Yuning Mao, Yu Meng, Meng Qu, Xiang Ren, Jingbo Shang, Jiaming Shen, Yizhou Sun, Fangbo Tao, Wenzhu Tong, Chenguang Wang, Qi Wang, Jingjing Wang, Xuan Wang, Ellen Wu, Jinfeng Xiao, Doris Xin, Frank Xu, Carl Yang, Zhijun Yin, Hongkun Yu, Xiao Yu, Quan Yuan, Keyang Zhang, Naijing Zhang, Xinyang Zhang, Yu Zhang, Shi Zhi, Qi Zhu, and Wanzheng Zhu.

I am deeply grateful to the other researchers I had the honor of working with through collaborations outside of UIUC or from the diverse and rewarding internship opportunities at LinkedIn, Snap Research, and Facebook, Shaunak Chatterjee, Xi Chen, Wei Cheng, Bruce Deng, Souvik Ghosh, Tiangao Gou, Fangqiu Han, Xinran He, How Jing, Lance Kaplan, Liangyue Li, David Liem, Roger Jie Luo, Shaoliang Nie, Romer Rosales, Martin Saveski, Mitul Tiwari, Matthew Walker, Wei Wang, Xin Fu, and Yichao Zhou.

Lastly and above all, I owe my deepest gratitude to my family who have provided unconditional love and support. This dissertation is dedicated to them.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

Objects interconnected with each other can be represented by networks. Examples of networks include social networks, bibliographic networks, biomedical networks, traffic networks, *etc.* In real-world scenarios, networks often contain heterogeneity due to the heterogeneous nature of the world [1, 2].

Ostensibly the heterogeneity of real-world network appears as the typed essence of nodes and edges. The most commonly seen typed networks are the *multi-view network* and the more general *heterogeneous information network* (HIN). A multi-view network [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]. consists of multiple network views, where each view corresponds to a type of edge, and all views share the same set of nodes with the same node type. For example, a multi-view network in ecology can be used to represent the relation among species, where each node stands for a species, and the six views represent predation, competition, symbiosis, parasitism, protocooperation, and commensalism, respectively. On social networking services, a four-view network upon users can be used to describe the widely seen social relationship and interaction including friendship, following, message exchange, and post viewing. The more general concept of heterogeneous information networks (HINs) [2, 14, 15, 16, 17] additionally involve different types of nodes. The bibliographical information network is a typical example, where researchers, papers, organizations, and publication venues are interrelated, and the IMDb network is another HIN containing information about users' preferences over movies and have five different node types: user, movie, actor, director, and genre.

In my practice of mining real-world networks, I discover that the *heterogeneous association* among network components is an intrinsic and important aspect of these networks. Being able to harness such heterogeneous association can benefit many real-world applications.

Association Among Typed Network Components

Beyond the low-level heterogeneity in that nodes and edges have their specific types, the node types and the edge types also naturally induce higher-level typed network components. For illustration, let us consider the following examples.

Example 1.1. ***Type constrained random walks.*** *Random walks have long been leveraged as a tool to analyze and mine networks. In the context of networks with typed nodes and typed edges, a type constrained random walk is sampled under a specific constraint that specifies the types of nodes and edge along the walk. The set of all random walks sampled under the same constraint can be one network component. Different components defined as such may reflect different semantic facets of the network.*

Example 1.2. ***Meta-paths.*** *A meta-path is a concatenation of node types linked by edge types. Namely, meta-paths can be used to categorize path instances in a network. In a bibliographic network, the meta-path [author]–[paper]–[author] reveals co-authorship, while [author]–[paper]–[venue]–[paper]–[author] represents two authors publish papers in the same venue. Each meta-path can correspond to one network component.*

Example 1.3. ***Induced subnetworks.*** *A subnetwork can be induced by one node type or one edge type, where nodes or edges of other types are filtered out from the original network. For instance, in a social network, the subnetwork induced by edge type friendship would be a homogeneous network over user nodes concerning only the friendship connections. Each subnetwork can be a network component.*

While these network components are the natural outcome of heterogeneity in networks, the association across these components often reflect critical and intrinsic information of the concerned real-world networks. Some components may contain compatible semantics, and joint modeling them usually yield improved performance. Meanwhile, some components may generate completely conflicting results in certain tasks (*e.g.*, network embedding) because they embody incompatible semantics. Naively incorporating more components in a model not able to handle such incompatibility may not always benefit a network mining task. That is, the heterogeneity of real-world networks also lie in the association across different network components.

In this dissertation, I study the prevalence of the heterogeneous association and its implication to network mining tasks. Methods and principles are also proposed to overcome accompanying challenges or, better still, exploit such heterogeneous association.

Association Strength Among Network Components Varies by Dataset

Network embedding has emerged as a scalable representation learning method that generates distributed node representations for networked data [18, 19, 20, 21]. Specifically, network embedding projects networks into embedding spaces, where nodes are represented by embedding vectors. With the semantic information of each node encoded, these vectors can be directly used as node features in various downstream applications [18, 19, 20].

In our practice of embedding multi-view networks, we identify that the association among network views obviously differ in different datasets. In some datasets, edges between the same pair of nodes may be observed in different views due to shared latent reasons. For instance, in a social network, if we observe an edge between a user pair in either the message exchange view or the post viewing view, likely these two users are happy to be associated with each other. In such a scenario, these views may complement each other, and embedding

them jointly may potentially yield better results than embedding them independently. We call such synergetic effect in jointly embedding multiple views by *collaboration*. On the other hand, it is possible for different network views to have different semantic meanings; it is also possible that a portion of nodes have completely disagreeing edges in different views since edges in different views are formed due to distinct latent reasons. For example, a professional relationship may not always align well with friendship. If we embed the profession view and the friendship view into the same metric space, the unique information carried by different network views would be lost. We refer to such need for preserving unique information carried by different views as *preservation*. Embedding algorithms catering for only *collaboration* or *preservation* achieve drastically different performance in different datasets as the association strength among network views varies by dataset.

Heterogeneity Exists in Association Strength within One Network

A fundamental problem in HIN analysis is to define proper measures to characterize the relevance between node pairs in the network, which also benefits various downstream applications, such as similarity search, recommendation, and community detection [2, 14]. We dive into the investigation of heterogeneous association within one network in the context of this fundamental problem.

Most existing studies derive their HIN relevance measures on the basis of *meta-path* [2, 14, 22], which is defined as a concatenation of multiple node types linked by corresponding edge types. Based on the concept of *meta-path*, researchers have proposed PathCount, PathSim [22], and path constrained random walk [23] to measure relevance between node pairs. On top of these studies, people have explored the ideas of incorporating richer information [24, 25] and more complex typed structures [26, 27, 28] to define more effective relevance scoring functions, or adding supervision to derive task-specific relevance measures [29, 30, 31]. We identify that the association strength across different meta-paths is not uniform. Taking such a heterogeneous association into account yields a more reliable and interpretable relevance measure for HINs.

Harnessing Heterogeneous Association by Identifying Aspects of an HIN

The heterogeneity in HINs poses a specific challenge for the problem of learning embedding in HINs. There are multiple attempts in studying HIN embedding or tackling specific application tasks using HIN embedding [32, 29, 33, 34]. Though these studies formulate the problem differently with respective optimization objectives, they share a similar underlining philosophy: using a unified objective function to embed all the nodes into *one* space.

However, embedding all the nodes into one metric space may lead to information loss

because different components with weak association strength may carry distinct semantic meaning despite being in the same network. To alleviate this problem, we propose a simple yet effective solution: first group network components with strong association into the same aspects in an unsupervised fashion, and then mine each aspect separately.

Harnessing Heterogeneous Association with Heterogeneous Metrics

In the face of those above potentially incompatible semantics in HINs embedding, we additionally observe different extents of such incompatibility. As a result, it can be expected that an algorithm would generate better embeddings if it additionally models such subtlety in semantic incompatibility.

Instead of hard-partitioning networks into multiple aspects, we propose to model such incompatibility with heterogeneous metrics. In this way, an HIN can be transcribed into its embedding representation as comprehensively as possible. This approach also provides an easy-to-use approach to unleash the power of HINs in a wide variety of applications with no expertise or supervision required in the embedding learning process, because with HINs comprehensives transcribed, one can again pipe the unsupervisedly learned embeddings to off-the-shelf machine learning algorithms for a wide range of applications.

Exploiting Heterogeneous Association Among Nodes and Contexts with an Application to Hypernymy Discovery

The context is widely used as an important facet for characterizing each node in the study of networks. In a typed network, particularly an HIN, we argue there exist multiple natural approaches to define the context. For instance, all the nodes adjacent to a particular node can be defined as the context of this node. Alternatively, one may treat all nodes reachable via a certain meta-path as the context of the original node. As such, the contexts with different definitions in an HIN can be viewed as different components of the network from a perspective orthogonal to the previous examples, and we observe that in the association strength among nodes and different contexts can vary drastically in a typed network.

Particularly, we apply the intention of exploiting such heterogeneous association among nodes and contexts in the task of hypernymy discovery from networks, where the hypernymy relation is primarily discovered by the distributional inclusion hypothesis (DIH). We demonstrate that different nodes have varied compatibility with different contexts when invoking DIH, and with the heterogeneous association modeled, the approach of hypernymy discovery from networks enjoys a greater range of successful deployment.

Organization

The remainder of this dissertation is organized as follows. In Chapter 2, we provide a

comprehensive survey on typed real-world networks and the mining tasks involved in this dissertation. In Chapter 3 and Chapter 4, using real-world datasets, we present an in-depth analysis of the prevalence and the impact of heterogeneous association across networks and within each network. In Chapter 5 and Chapter 6, we lay out two approaches that can be used to overcome the challenges introduced by heterogeneous association using embedding as example task. In Chapter 7, we apply the intention of modeling heterogeneous association to the task of hypernymy discovery from text-rich heterogeneous information networks, where the task of hypernymy discovery is an important and fundamental task in the field of text mining and natural language processing. Finally, Chapter 9 concludes the dissertation.

CHAPTER 2: LITERATURE REVIEW

In this chapter, we survey existing studies that are related to typed real-world networks and the mining tasks involved in this dissertation.

2.1 TYPED REAL-WORLD NETWORKS

The majority of existing methods for multi-view networks aim to bring performance boost in traditional tasks, such as clustering [4, 6], classification [7], and dense subgraph mining [10]. Another line of research focuses on measuring and analyzing cross-view interrelations in multi-view networks [35, 36, 37, 13], but they do not discuss the characteristics of embedding multi-view networks, nor do they study how their proposed measures and analyses can relate to the embedding learning of multi-view networks.

Heterogeneous information network (HIN) has been extensively studied as a powerful and effective paradigm to model networked data with rich and informative type information [2, 14]. Following this paradigm, a great many applications such as classification, clustering, recommendation, and outlier detection have been studied [2, 38, 14, 15, 17, 16]. However, many of these existing works rely on feature engineering [15, 17, 16]. Meanwhile, we aim at proposing an unsupervised feature learning method for general HINs that can serve as the basis for different downstream applications.

2.2 HOMOGENEOUS NETWORK EMBEDDING

Network embedding has emerged as an efficient and effective representation learning approach for networked data [18, 39, 19, 40, 20, 21, 41, 42, 43, 40], which significantly spares the labor and sources in transforming networks into features that are more machine-actionable. Early network embedding algorithms start from handling the simple, homogeneous networks, and many of them trace to the skip-gram model [44] that aims to learn word representations where words with similar context have similar representation [18, 19, 40, 20]. Besides skip-gram, algorithms for preserving certain other homogeneous network properties have also been studied [39, 21, 45, 46, 47, 48]. The use of edge representations for homogeneous network embedding is discussed in a recent work [49], but such edge representations are designed to distinguish the direction of an edge, instead of encoding richer semantics such as edge type in our case.

2.3 NETWORK EMBEDDING IN TYPED NETWORKS

An attention-based collaboration framework is recently proposed for multi-view network embedding [50]. The problem setting of this work differs from ours since it requires supervision for its attention mechanism. Besides, this approach does not directly model *preservation*, since the final embedding derived via linear combination in this framework is a trade off between representations from all views. A deep learning architecture has also been proposed for embedding multi-networks [51], where the multi-network is a more general concept than the multi-view network and allows many-to-many correspondence across networks. While the proposed model can be applied to the more specific multi-view networks, it does not focus on the study of the characteristics of multi-view network embedding. Another group of related work studies the problem of jointly modeling multiple network views using latent space models [52, 53]. These works again do not model *preservation*. In a bigger scope, a couple of studies have recently been conducted in embedding more general networks, such as heterogeneous information networks (HINs) [34, 32, 54, 55]. Some of them build the algorithms on top of meta-paths [54, 55], but meta-paths are usually more meaningful for HINs with multiple node types than multi-view networks. Some other methods are designed for networks with additional side information [32, 42] or particular structures [34, 33], which do not apply to multi-view networks. Most importantly, these methods are designed for networks they intend to embed, and therefore do not focus on the study of the particular needs and characteristics for embedding multi-view networks.

Heterogeneous information network (HIN) has been extensively studied since the past decade for its ubiquity in real-world data and efficacy in fulfilling tasks, such as classification, clustering, recommendation, and outlier detection [2, 14, 15, 17, 16]. To marry the advantages of HIN and network embedding, a couple of algorithms have been proposed very recently for embedding learning in heterogeneous information networks [56, 55, 54, 32, 33, 34, 57]. One line of work first uses human expertise or supervision to select meta-paths for a given task or limit the scope of candidate meta-paths, and then proposes methods to transfer the semantics encoded in meta-paths to the learned embedding [56, 55, 54]. While this direction has been showed to be effective in solving problems that fit the semantics of the chosen meta-paths, it differs from the research scope of ours because they mostly focus on providing quality representations for downstream tasks concerning the node types on the two ends of chosen meta-paths, while we aim at developing methods to transcribe the entire HIN to embeddings as comprehensively as possible. Beyond meta-paths, some approaches have been proposed to embed specific kinds of HINs [33, 34] with specific objectives such as representing event data or learning predictive text embeddings. Some other approaches

study HINs with additional side information [32] that cannot be generalized to all HINs. Besides, all of these approaches embed the input HIN into only one metric space. Embedding in the context of HIN has also been studied for tasks with additional supervision [29, 58, 59]. These methods either yield features specific to given tasks, and are outside of the scope of unsupervised HIN embedding that we study.

Multi-sense embedding is related to our problem since it exploits the heterogeneity in word senses. The idea of multiple aspects is in a way related to the polysemy of words. There have been some studies on inferring multi-sense embeddings of words [60, 61, 62, 63], which aims at inferring multiple embedding vectors for each word. However, this task differs from ours in the following perspectives. Firstly, each node may have multiple embeddings due to the semantic subtlety associated with each aspect; while in multi-sense word embedding learning, the number of senses for each word varies. Secondly, we aim at studying the embedding in HINs; while multi-sense embeddings word embedding learning is for textual data. Therefore, the methods developed for multi-sense embedding learning cannot be directly applied to the task of HIN embedding learning with aspects.

2.4 RELEVANCE MEASURE IN HETEROGENEOUS INFORMATION NETWORKS

The problem of deriving relevance between node pairs has been extensively studied for homogeneous information networks. Relevance measures of this type include the random walk based *Personalized PageRank* and *SimRank* [64], the neighbor-based *common neighbors* and *Jaccard's coefficient*, the path-based *Katz* [65], *etc.* To generalize relevance from the homogeneous networks to the typed heterogeneous case, researchers have been exploring from multiple perspectives. One perspective, as in *PathCount* and *PathSim* from [22] and *Path-Constrained Random Walk* from [23], is to first compute relevance score along each meta-path, and then glue scores from all types together via linear combination to establish the composite measure. A great many applications [66, 2, 14, 17, 16] based on this meta-path paradigm with linear combination have been proposed. Our proposed method follows this meta-path paradigm, but goes beyond linear combination to model *cross-meta-path synergy* that we have observed from real-world HINs. Another perspective is to go beyond meta-path and derive relevance based on the more complex graph structures [26, 27]. While these approaches can yield good performance, they differ from our proposed methods for further entailing label information or expertise in designing graph structure. Also, they do not carry probabilistic interpretations. Besides, people have explored the idea of incorporating richer information [24, 25] to define more effective relevance scoring functions, or adding supervision to derive task-specific relevance measures [29, 30, 31]. While being valuable,

these works are out of the scope of the problem we study in our work, where we address the basic, unsupervised case with no additional information as our starting point of studying HIN relevance from the probabilistic perspective.

2.5 HYPERNYMY DISCOVERY

In this dissertation, we also dive into the problem of hypernymy discovery from text-rich heterogeneous information networks as an application scenario of modeling heterogeneous association. The task of hypernymy discovery is an important and fundamental task in the field of text mining and natural language processing.

Distributional methods constitute one major line of research for hypernymy discovery [67, 68] and can be adapted to hypernymy discovery from network data. Early studies proposed symmetric distributional measures for hypernymy discovery that only capture relevance between terms [69]. Whereafter, researchers have heavily investigated into asymmetric measures based on the *distributional inclusion hypothesis* (DIH) to comply with the asymmetricity of hypernymy relation [67, 70, 71]. Examples of popular DIH measures include WeedsPrec [72], APinc and balAPinc [73], ClarkeDE [74], cosWeeds, invCL [75], and WeightedCosine [76].

Closely related to our effort in rectifying the DIH by modeling context granularity, several studies have also studied the validity of the DIH. These studies have also suggested the DIH may not always hold accurate and proposed solutions orthogonal to ours [77, 78, 79]. Santus et al. [77] propose an entropy-based measure SLQS that do not rely on the DIH, while some other studies suggested only certain units in the context should be used to generate features [78, 79]. We note that these approaches do not contradict with ours, because they are all based on the default context granularity, while we argue that DIH would hold at proper context granularities for each hypernym-hyponym pair.

Hearst, *et al.* [80] pioneered the line of pattern-based hypernymy discovery methods from text corpus using several hand-crafted lexico-syntactic patterns, and a substantial number of methods have been proposed following this idea [81, 82, 83]. It has also been argued that Hearst pattern based methods tend to achieve high precision with compromised recall [67, 81, 82, 84, 85]. Attempts have also been made to further improve the recall [86, 87, 88, 89] and the precision [81, 90, 82, 88, 91, 86, 92]. In our framework, we use the straightforward Hearst pattern based method to extract weak supervision pairs in the hope of yielding pairs with decent precision and without much additional engineering.

With additional supervision available, researchers have proposed and trained models to infer hypernymy based on the representations of a term pair [92, 93, 94, 78, 95, 96, 97, 98].

Methods for deriving such representations include the aforementioned pattern-based methods and distributional methods as well as the compact, distributed representations generated from models such as word2vec [44], GloVe [99], ivLBL [100], and SensEmbed [101].

Lastly, we comment that the benchmark datasets used in the above hypernymy discovery related work do not apply to our problem setting, since the input data in these benchmarks are purely corpus without the availability of related network data.

CHAPTER 3: ASSOCIATION STRENGTH AMONG NETWORK COMPONENTS VARIES BY DATASET

3.1 OVERVIEW

In real-world applications, objects can be associated with different types of relations. These objects and their relationships can be naturally represented by multi-view networks, which are also known as multiplex networks or multi-view graphs [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]. As shown in Figure 3.1a, a multi-view network consists of multiple network views, where each view corresponds to a type of edge, and all views share the same set of nodes. In ecology, a multi-view network can be used to represent the relation among species, where each node stands for a species, and the six views represent predation, competition, symbiosis, parasitism, protocooperation, and commensalism, respectively. On social networking services, a four-view network upon users can be used to describe the widely seen social relationship and interaction including friendship, following, message exchange, and post viewing. With such vast availability of multi-view networks, one could be interested in extracting knowledge or business value from data. In order to achieve this goal with the progressively developed computing power, it is of interest to first transform the multi-view networks into a different form of representations that are more machine actionable.

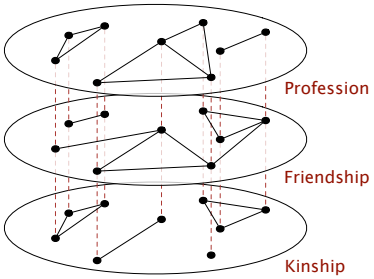
Network embedding has emerged as a scalable representation learning method that generates distributed node representations for networked data [18, 19, 20, 21]. Specifically, network embedding projects networks into embedding spaces, where nodes are represented by embedding vectors. With the semantic information of each node encoded, these vectors can be directly used as node features in various downstream applications [18, 19, 20]. Motivated by the success of network embedding in representing homogeneous networks [18, 19, 20, 21, 102, 39], where nodes and edges are untyped, we believe it is important to study the problem of embedding multi-view networks.

To design embedding algorithms for multi-view networks, the major challenge lies in how to make use of the type information on edges from different views. As a result, we are interested in investigating into the following two problems:

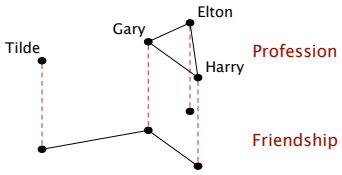
1. With the availability of multiple edge types, what are the characteristics that are *specific* and *important* to multi-view network embedding?

2. Can we achieve better embedding quality by modeling these characteristics jointly?

To answer the first problem, we identify two characteristics, *preservation* and *collaboration*, from our practice of embedding real-world multi-view networks. We describe the concepts of



(a) A toy multi-view network.



(b) A close-up look of profession view and friendship view.

Figure 3.1: A toy example of multi-view networks where each node represents a person and the three views correspond to three types of interpersonal relations. Co-workers are linked in the profession view, friends are linked in the friendship view, and relatives are linked in the kinship view.

preservation and *collaboration* as follows. **Collaboration** – In some datasets, edges between the same pair of nodes may be observed in different views due to shared latent reasons. For instance, in a social network, if we observe an edge between a user pair in either the message exchange view or the post viewing view, likely these two users are happy to be associated with each other. In such scenario, these views may complement each other, and embedding them jointly may potentially yield better results than embedding them independently. We call such synergetic effect in jointly embedding multiple views by *collaboration*. The feasibility of enjoying this synergetic effect is also the main intuition behind most existing multi-view network algorithms [3, 4, 5, 6, 7, 8, 9, 10, 11]. **Preservation** – On the other hand, it is possible for different network views to have different semantic meanings; it is also possible that a portion of nodes have completely disagreeing edges in different views since edges in different views are formed due to distinct latent reasons. For example, professional relationship may not always align well with friendship. If we embed the profession view and the friendship view in Figure 3.1b into the same embedding space, the embedding of Gary will be close to both Tilde and Elton. As a result, the embedding of Tilde will also not be too distant from Elton due to transitivity. However, this is not a desirable result, because Tilde and Elton are not closely related in terms of either profession or friendship according to the original multi-view network. In other words, embedding in this way fails to preserve the unique information carried by different network views. We refer to such need for preserving unique information carried by different views as *preservation*. The detailed discussion of the presence and importance of *preservation* and *collaboration* is presented in Section 3.3.

Furthermore, it is also possible for *preservation* and *collaboration* to co-exist in the same multi-view network. Two scenarios can result in this situation: (i) a pair of views are generated from very similar latent reason, while another pair of views carries completely different semantic meanings; and more subtly (ii) for the same pair of views, one portion of nodes have consistent edges in different views, while another portion of nodes have totally disagreeing edges in different views. One example of the latter scenario is that professional relationship does not align well with friendship in some cultures, whereas co-workers often become friends in certain other cultures [103]. Therefore, we are also interested in exploring the feasibility of achieving better embedding quality by modeling *preservation* and *collaboration* simultaneously, and we address this problem in Section 3.4 and beyond.

We summarize our contributions as follows. (i) We propose to study the characteristics that are specific and important to multi-view network embedding, and identify *preservation* and *collaboration* as two such characteristics from the practice of embedding real-world multi-view networks. (ii) We explore the feasibility of attaining better embedding by simultaneously modeling *preservation* and *collaboration*, and propose two multi-view network embedding methods – MVN2VEC-CON and MVN2VEC-REG. (iii) We conduct experiments with various downstream applications on a series of synthetic datasets and three real-world multi-view networks, including an internal dataset sampled from the Snapchat social network. These experiments corroborate the presence and importance of *preservation* and *collaboration*, and demonstrate the effectiveness of the proposed methods.

3.2 PRELIMINARIES

Definition 3.1 (Multi-View Network)**.** *A multi-view network $G = (\mathcal{U}, \{\mathcal{E}^{(v)}\}_{v\in\mathcal{V}})$ is a network consisting of a set $\mathcal{U}$ of nodes and a set $\mathcal{V}$ of views, where $\mathcal{E}^{(v)}$ consists of all edges in view $v \in \mathcal{V}$. If a multi-view network is weighted, then there exists a weight mapping $w : \{\mathcal{E}^{(v)}\}_{v\in\mathcal{V}} \to \mathbb{R}$ such that $w^{(v)}_{uu'} := w(e^{(v)}_{uu'})$ is the weight of the edge $e^{(v)}_{uu'} \in \mathcal{E}^{(v)}$, which joints nodes $u \in \mathcal{U}$ and $u' \in \mathcal{U}$ in view $v \in \mathcal{V}$.*

Additionally, when context is clear, we use the network view v of multi-view network $G = (\mathcal{U}, \{\mathcal{E}^{(v)}\}_{v\in\mathcal{V}})$ to denote the untyped network $G^{(v)} = (\mathcal{U}, \mathcal{E}^{(v)})$.

Definition 3.2 (Network Embedding)**.** *Network embedding aims at learning a (center) embedding $\mathbf{f}_u \in \mathbb{R}^D$ for each node $u \in \mathcal{U}$ in a network, where $D \in \mathbb{N}$ is the dimension of the embedding space.*

Besides the center embedding $\mathbf{f}_u \in \mathbb{R}^D$, a family of popular algorithms [44, 20] also deploy a context embedding $\tilde{\mathbf{f}}_u \in \mathbb{R}^D$ for each node u. Moreover, when the learned embedding is

Table 3.1: Summary of symbols

Symbol	Definition
$\mathcal{V}$	The set of all network views
$\mathcal{U}$	The set of all nodes
$\mathcal{E}^{(v)}$	The set of all edges in view $v \in \mathcal{V}$
$\mathcal{W}^{(v)}$	The list of random walk pairs from view $v \in \mathcal{V}$
$\mathbf{f}_u$	The final embedding of node $u \in \mathcal{U}$
$\mathbf{f}_u^v$	The center embedding of node $u \in \mathcal{U}$ *w.r.t.* view $v \in \mathcal{V}$
$\tilde{\mathbf{f}}_u^v$	The context embedding of node $u \in \mathcal{U}$ *w.r.t.* view $v \in \mathcal{V}$
$\theta \in [0, 1]$	The hyperparameter on parameter sharing in MVN2VEC-CON
$\gamma \in \mathbb{R}_{\geq 0}$	The hyperparameter on regularization in MVN2VEC-REG
$D \in \mathbb{N}$	The dimension of the embedding space

used as the feature vector for downstream applications, we take the center embedding of each node as feature following the common practice in algorithms involving context embedding.

3.3 PRESERVATION AND COLLABORATION IN MULTI-VIEW NETWORK EMBEDDING

In this section, we elaborate on the intuition and presence of *preservation* and *collaboration* – the two characteristics that we have introduced in Chapter 1 and deem important for multi-view network embedding. In particular, we first describe and investigate the motivating phenomena that are observed in the practice of embedding real-world multi-view networks. Then, we discuss how they can be explained by the two proposed characteristics.

Two straightforward approaches for embedding multi-view networks. Most existing network embedding methods [18, 19, 20, 21, 102, 39] are designed for homogeneous networks, where nodes and edges are untyped, while we are interested in studying the problem of embedding multi-view networks. To extend any untyped network embedding algorithm to multi-view networks, two straightforward yet practical approaches exist. We refer to these two approaches as the *independent* model and the *one-space* model.

Using any untyped network embedding method, we denote $\mathbf{f}_u^v \in \mathbb{R}^{d(v)}$ the (center) embedding of node $u \in \mathcal{U}$ achieved by embedding only the view $v \in \mathcal{V}$ of the multi-view network, where $d(v)$ is the dimension of the embedding space for network view v. With such notation, the *independent* model and the *one-space* model are given as follows.

- **Independent**. Embed each view independently, and then concatenate to derive the final embedding $\mathbf{f}_u$. That is,

$$\mathbf{f}_u = \bigoplus_{v \in \mathcal{V}} \mathbf{f}_u^v \in \mathbb{R}^D, \tag{3.1}$$

  where $D = \sum_{v \in \mathcal{V}} d(v)$, and $\bigoplus$ represents concatenation. In other words, the embedding of each node in the *independent* model resides in the direct sum of multiple

Table 3.2: Embedding quality of two real-world multi-view networks using the *independent* model and the *one-space* model.

Dataset	Metric	*independent*	*one-space*
YouTube	ROC-AUC	**0.931**	0.914
	AUPRC	**0.745**	0.702
Twitter	ROC-AUC	0.724	**0.737**
	AUPRC	0.447	**0.466**

embedding spaces. This approach preserves the information embodied in each view, but do not allow collaboration across different views in the embedding learning process.

- **One-space**. Let the embedding for different views to share parameters when learning the final embedding $\mathbf{f}_u$. That is,

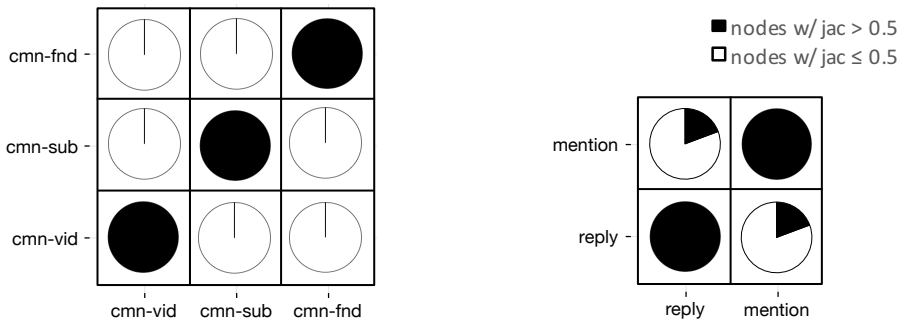
$$\mathbf{f}_u = \mathbf{f}_u^v \in \mathbb{R}^D, \ \forall v \in \mathcal{V}, \tag{3.2}$$

where $D = d(v)$ for all $v \in \mathcal{V}$. In other words, each dimension of the final embedding space correlates with all views of the concerned multi-view network. This approach enables different views to collaborate in learning a unified embedding, but do not preserve information specifically carried by each view. This property of the *one-space* model is corroborated by experiment presented in Section 3.5.5.

In either of the above two approaches, the same treatment to the center embedding is applied to the context embedding when applicable. It is also worth noting that the embedding learned by the *one-space* model cannot be obtained by linearly combining $\{\mathbf{f}_u^v\}_{v\in\mathcal{V}}$ in the *independent* model. This is because most network embedding models are non-linear models.

Embedding real-word multi-view networks by straightforward approaches. In our work, *independent* and *one-space* are implemented on top of a random walk plus skip-gram approach as widely seen in the literature [18, 19, 102]. The experiment setup and results are concisely introduced at this point, while detailed description of algorithm, datasets, and more comprehensive experiment results are deferred to Section 3.4 and 3.5. Two networks, YouTube and Twitter, are used in these exploratory experiments with users being nodes on each network. YouTube has three views representing common videos (cmn-vid), common subscribers (cmn-sub), and common friends (cmn-fnd) shared by each pair of users, while Twitter has two views corresponding to replying (reply) and mentioning (mention) among users. The downstream evaluation task is to infer whether two users are friends, and the results are presented in Table 3.2.

It can be seen that the *independent* model consistently outperformed the *one-space* model



(a) YouTube. **(b) Twitter.**

Figure 3.2: Agreement between information carried by each pair of network views given by a Jaccard coefficient based measurement.

in the YouTube experiment, while the *one-space* model outperformed the *independent* model in Twitter. These exploratory experiments make it clear that neither of the two straightforward approaches is categorically superior to the other. Furthermore, we interpret the varied performance of the two approaches by the varied extent of needs for modeling *preservation* and modeling *collaboration* when embedding different networks. Specifically, recall that the *independent* model only captures *preservation*, while *one-space* only captures *collaboration*. As a result, we speculate if a certain dataset craves for more *preservation* than *collaboration*, the *independent* model would outperform the *one-space* model, otherwise, the *one-space* model would win.

In order to corroborate our interpretation of the results, we further examine the involved datasets, and look into the agreement between information carried by different network views. We achieve this by a Jaccard coefficient–based measurement, where the Jaccard coefficient is a similarity measure with range $[0, 1]$, defined as $J(\mathcal{S}_1, \mathcal{S}_2) = |\mathcal{S}_1 \cap \mathcal{S}_2|/|\mathcal{S}_1 \cup \mathcal{S}_2|$ for set $\mathcal{S}_1$ and set $\mathcal{S}_2$. Given a pair of views in a multi-view network, a node can be connected to a different set of neighbors in each of the two network views. The Jaccard coefficient between these two sets of neighbors can then be calculated. In Figure 3.2, we apply this measurement on the YouTube dataset and the Twitter data, respectively, and illustrate the proportion of nodes with the Jaccard coefficient greater than 0.5 for each pair of views.

As presented in Figure 3.2, little agreement exists between each pair of different views in YouTube. As a result, it is not surprising that *collaboration* among different views is not as needed as *preservation* in the embedding learning process. On the other hand, a substantial portion of nodes have Jaccard coefficient greater than 0.5 over different views in the Twitter dataset. It is therefore also not surprising to see modeling *collaboration* brings about more benefits than modeling *preservation* in this case.

3.4 THE MVN2VEC MODELS

In the previous section, *preservation* and *collaboration* are identified as important characteristics for multi-view network embedding. In the extreme cases, where only *preservation* is needed – each view carries a distinct semantic meaning – or only *collaboration* is needed – all views carry the same semantic meaning – it is advisable to choose between *independent* and *one-space* to embed a multi-view network. However, it is of interest to study the also likely scenario where both *preservation* and *collaboration* co-exist in given multi-view networks. Therefore, we are motivated to explore the feasibility of achieving better embedding by simultaneously modeling both characteristics. To this end, we propose and experiment with two approaches that capture both characteristics, without over-complicating the model or requiring additional supervision. These two approaches are named MVN2VEC-CON and MVN2VEC-REG, where MVN2VEC is short for **m**ulti-**v**iew **n**etwork **to vec**tor, while CON and REG stand for constrained and regularized, respectively.

As with the notation convention in Section 3.3, we denote $\mathbf{f}_u^v \in \mathbb{R}^{d(v)}$ and $\tilde{\mathbf{f}}_u^v \in \mathbb{R}^{d(v)}$ the center and context embedding, respectively, of node $u \in \mathcal{U}$ for view $v \in \mathcal{V}$. Further given the network view v, *i.e.*, $G^{(v)} = (\mathcal{U}, \mathcal{E}^{(v)})$, we use an intra-view loss function to measure how well the current embedding can represent the original network view

$$l(\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u\in\mathcal{U}} | G^{(v)}). \tag{3.3}$$

We defer the detailed definition of this loss function (Eq. (3.3)) to a later point of this section. Moreover, we let $d(v) = D/|\mathcal{V}| \in \mathbb{N}$ for all $v \in \mathcal{V}$ out of convenience for model design. To further incorporate multiple views with the intention to model both *preservation* and *collaboration*, two approaches are proposed as follows.

mvn2vec-con. The MVN2VEC-CON model does not enforce further design on the center embedding $\{\mathbf{f}_u^v\}_{u\in\mathcal{U}}$ in the hope of preserving the semantics of each individual view. To reflect *collaboration*, MVN2VEC-CON includes further constraints on the context embedding for parameter sharing across different views

$$\tilde{\mathbf{f}}_u^v = \varphi_\theta^v(\{\tilde{\mathbf{g}}_u^{v'}\}_{v'\in\mathcal{V}}) \coloneqq (1-\theta)\cdot\tilde{\mathbf{g}}_u^v + \frac{\theta}{|\mathcal{V}|}\cdot\sum_{v'\in\mathcal{V}}\tilde{\mathbf{g}}_u^{v'}, \tag{3.4}$$

where $\theta \in [0, 1]$ is a hyperparameter controlling the extend to which model parameters are shared. The greater the value of θ, the more the model enforces parameter sharing and thereby encouraging more *collaboration* across different views. This design aims at allowing different views to collaborate by passing information via the shared parameters in

the embedding learning process. That is, the MVN2VEC-CON model solves the following optimization problem

$$\min_{\{\mathbf{f}_u^v, \tilde{\mathbf{g}}_u^v\}_{u\in\mathcal{U}, v\in\mathcal{V}}} \sum_{v\in\mathcal{V}} l(\{\mathbf{f}_u^v, \varphi_\theta^v(\{\tilde{\mathbf{g}}_u^{v'}\}_{v'\in\mathcal{V}})\}_{u\in\mathcal{U}} | G^{(v)}), \tag{3.5}$$

where $\varphi_\theta^v(\{\tilde{\mathbf{g}}_u^{v'}\}_{v'\in\mathcal{V}})$ is defined in Eq. (3.4). After model learning, the final embedding for node u is given by $\mathbf{f}_u = \bigoplus_{v\in\mathcal{V}} \mathbf{f}_u^v$. We note that in the extreme case when θ is set to be 0, the model will be identical to the *independent* model discussed in Section 3.3.

mvn2vec-reg. In stead of setting hard constraints on how parameters are shared across different views, the MVN2VEC-REG model regularizes the embedding across different views and solves the following optimization problem

$$\min_{\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u\in\mathcal{U}, v\in\mathcal{V}}} \sum_{v\in\mathcal{V}} l(\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u\in\mathcal{U}} | G^{(v)}) + \gamma \cdot [\mathcal{R}^v + \tilde{\mathcal{R}}^v], \tag{3.6}$$

where $\|\cdot\|_2$ is the l-2 norm, $\mathcal{R}^v = \sum_{u\in\mathcal{U}} \left\| \mathbf{f}_u^v - \frac{1}{|\mathcal{V}|} \sum_{v'\in\mathcal{V}} \mathbf{f}_u^{v'} \right\|_2^2$, $\tilde{\mathcal{R}}^v = \sum_{u\in\mathcal{U}} \left\| \tilde{\mathbf{f}}_u^v - \frac{1}{|\mathcal{V}|} \sum_{v'\in\mathcal{V}} \tilde{\mathbf{f}}_u^{v'} \right\|_2^2$, and $\gamma \in \mathbb{R}_{\geq 0}$ is a hyperparameter. This model captures *preservation* again by letting $\{\mathbf{f}_u^v\}_{u\in\mathcal{U}}$ and $\{\tilde{\mathbf{f}}_u^v\}_{u\in\mathcal{U}}$ to reside in the embedding subspace specific to view $v \in \mathcal{V}$, while each of these subspaces are distorted via cross-view regularization to model *collaboration*. Similar to the MVN2VEC-CON model, the greater the value of the hyperparameter γ, the more the *collaboration* is encouraged, and the model is identical to the *independent* model when $\gamma = 0$.

Intra-view loss function. There are many possible approaches to formulate the intra-view loss function in Eq. (3.3). In our framework, we adopt the random walk plus skip-gram approach, which is one of the most common methods used in the literature [18, 19, 102]. Specifically, for each view $v \in \mathcal{V}$, multiple rounds of random walks are sampled starting from each node in $G^{(v)} = (\mathcal{U}, \mathcal{E}^{(v)})$. Along any random walk, a node $u \in \mathcal{U}$ and a neighboring node $n \in \mathcal{U}$ constitute one random walk pair, and a list $\mathcal{W}^{(v)}$ of random walk pairs can thereby be derived. We defer the detailed description on the generation of $\mathcal{W}^{(v)}$ to a later point in this section. The intra-view function is then given by

$$l(\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u\in\mathcal{U}} | G^{(v)}) = -\sum_{(u,n)\in\mathcal{W}^{(v)}} \log p^{(v)}(n|u), \tag{3.7}$$

where

$$p^{(v)}(n|u) = \frac{\exp\left(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v\right)}{\sum_{n'\in\mathcal{U}} \exp\left(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n'}^v\right)}. \tag{3.8}$$

Algorithm 3.1: MVN2VEC-CON and MVN2VEC-REG

Input : the multi-view network $G = (\mathcal{U}, \{\mathcal{E}^{(v)}\}_{v\in\mathcal{V}})$ and the hyperparameters
Output: the final embedding $\{\mathbf{f}_u\}_{u\in\mathcal{U}}$

begin
  for $v \in \mathcal{V}$ **do**
    Sample a list $\mathcal{W}^{(v)}$ of random walk pairs
  Join and shuffle the lists of random walk pairs from all views to form a new random walk pair list $\mathcal{W}$
  for *each epoch* **do**
    `// The following for-loop is parallelized`
    for $(u, n) \in \mathcal{W}$ **do**
      if *using model* MVN2VEC-CON **then**
        Update $\{\mathbf{f}_u^v, \tilde{\mathbf{g}}_u^v\}_{u\in\mathcal{U},v\in\mathcal{V}}$ with one step descent using gradients in Eq. (3.9)–(3.11)
      if *using model* MVN2VEC-REG **then**
        Update $\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u\in\mathcal{U},v\in\mathcal{V}}$ with one step descent using gradients in Eq. (3.12)–(3.14)
  for $u \in \mathcal{U}$ **do**
    Derive the embedding for node u by $\mathbf{f}_u = \bigoplus_{v\in\mathcal{V}} \mathbf{f}_u^v$

Model inference. To optimize the objectives in Eq. (3.5) and (3.6), we opt to asynchronous stochastic gradient descent (ASGD) [104] following existing skip-gram–based algorithms [18, 19, 102, 20, 44]. In this regard, $\mathcal{W}^{(v)}$ from all views are joined and shuffled to form a new list $\mathcal{W}$ of random walk pairs for all views. Then each step of ASGD draws one random walk pair from $\mathcal{W}$, and updates corresponding model parameters with one-step gradient descent.

Moreover, due to the existence of partition function in Eq. (3.8), computing gradients of Eq. (3.5) and (3.6) is unaffordable with Eq. (3.7) being their parts. Negative sampling is hence adopted as in other skip-gram–based methods [18, 19, 102, 20, 44], which approximates $\log p^{(v)}(n|u)$ in Eq. (3.7) by

$$-\log\sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v) - \sum_{i=1}^{K} \mathbb{E}_{n_i'\sim P^{(v)}} \log\sigma(-\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n_i'}^v),$$

where $\sigma(x) = 1/(1+\exp(-x))$ is the sigmoid function, K is the negative sampling rate, $P^{(v)}(u) \propto \left[D_u^{(v)}\right]^{3/4}$ is the noise distribution, and $D_u^{(v)}$ is the number of occurrences of node u in $\mathcal{W}^{(v)}$ [44].

With negative sampling, the objective function involving one walk pair (u, n) drawn from

view v in MVN2VEC-CON is

$$\begin{aligned}\mathcal{O}_{\text{CON}} =& \log \sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_n^{v'}\}_{v' \in \mathcal{V}})) \\ &+ \sum_{i=1}^{K} \mathbb{E}_{n_i' \sim P^{(v)}} \log \sigma(-\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_{n_i'}^{v'}\}_{v' \in \mathcal{V}})).\end{aligned}$$

On the other hand, the objective function involving (u, n) from view v in MVN2VEC-REG is

$$\begin{aligned}\mathcal{O}_{\text{REG}} =& \log \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v) + \gamma \cdot (\mathcal{R}_u^v + \tilde{\mathcal{R}}_n^v) \\ &+ \sum_{i=1}^{K} \left[\mathbb{E}_{n_i' \sim P^{(v)}} \log \sigma(-\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n_i'}^v) + \gamma \cdot (\mathcal{R}_u^v + \tilde{\mathcal{R}}_{n_i'}^v) \right],\end{aligned}$$

and $\mathcal{R}_{\hat{u}}^v = \left\| \mathbf{f}_{\hat{u}}^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \mathbf{f}_{\hat{u}}^{v'} \right\|_2^2$, $\tilde{\mathcal{R}}_{\hat{u}}^v = \left\| \tilde{\mathbf{f}}_{\hat{u}}^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \tilde{\mathbf{f}}_{\hat{u}}^{v'} \right\|_2^2$.

We provide the gradients used for ASGD in the proposed algorithms as follows.

mvn2vec-con:

$$\begin{aligned}\frac{\partial \mathcal{O}_{\text{CON}}}{\partial \mathbf{f}_u^v} =& \left(1 - \sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_n^{v'}\}_{v' \in \mathcal{V}}))\right) \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_n^{v'}\}_{v' \in \mathcal{V}}) \\ &- \sum_{i=1}^{K} \sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_{n_i'}^{v'}\}_{v' \in \mathcal{V}})) \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_{n_i'}^{v'}\}_{v' \in \mathcal{V}}),\end{aligned} \tag{3.9}$$

$$\frac{\partial \mathcal{O}_{\text{CON}}}{\partial \tilde{\mathbf{g}}_n^{\hat{v}}} = \left(1 - \sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_n^{v'}\}_{v' \in \mathcal{V}}))\right) \cdot \mathbf{f}_u^v \cdot \begin{cases} \theta + \frac{1-\theta}{|\mathcal{V}|}, & \hat{v} = v, \\ \theta, & \hat{v} \neq v, \end{cases} \tag{3.10}$$

$$\frac{\partial \mathcal{O}_{\text{CON}}}{\partial \tilde{\mathbf{g}}_{n_i'}^{\hat{v}}} = -\sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_{n_i'}^{v'}\}_{v' \in \mathcal{V}})) \cdot \mathbf{f}_u^v \cdot \begin{cases} \theta + \frac{1-\theta}{|\mathcal{V}|}, & \hat{v} = v, \\ \theta, & \hat{v} \neq v. \end{cases} \tag{3.11}$$

mvn2vec-reg:

$$\begin{aligned}\frac{\partial \mathcal{O}_{\text{REG}}}{\partial \mathbf{f}_u^v} =& \left(1 - \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v)\right) \cdot \tilde{\mathbf{f}}_n^v - \sum_{i=1}^{K} \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n_i'}^v) \cdot \tilde{\mathbf{f}}_{n_i'}^v \\ &+ 2\gamma\Big(K + 1\Big)\Big(1 - \frac{1}{|\mathcal{V}|}\Big) \cdot \Big(\mathbf{f}_u^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \mathbf{f}_u^{v'}\Big),\end{aligned} \tag{3.12}$$

$$\frac{\partial \mathcal{O}_{\text{REG}}}{\partial \tilde{\mathbf{f}}_n^v} = \left(1 - \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v)\right) \cdot \mathbf{f}_u^v + 2\gamma\Big(1 - \frac{1}{|\mathcal{V}|}\Big) \cdot \Big(\tilde{\mathbf{f}}_n^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \tilde{\mathbf{f}}_n^{v'}\Big), \tag{3.13}$$

$$\frac{\partial \mathcal{O}_{\text{REG}}}{\partial \tilde{\mathbf{f}}^v_{n'_i}} = -\sigma(\mathbf{f}^v_u \cdot \tilde{\mathbf{f}}^v_{n'_i}) \cdot \mathbf{f}^v_u + 2\gamma\Big(1 - \frac{1}{|\mathcal{V}|}\Big) \cdot \Big(\tilde{\mathbf{f}}^v_{n'_i} - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \tilde{\mathbf{f}}^{v'}_{n'_i}\Big). \tag{3.14}$$

Note that in implementation, $|V|$ should be the number of views in which u is associated with at least one edge.

Random walk pair generation. Without additional supervision, we assume equal importance of different network views in learning embedding, and sample the same number $N \in \mathbb{N}$ of random walks from each view. To determine this number, we denote $n^{(v)}$ the number of nodes that are not isolated from the rest of the network in view $v \in \mathcal{V}$, $n_{max} \coloneqq \max\{n^{(v)} : v \in \mathcal{V}\}$, and let $N \coloneqq M \cdot n_{max}$, where M is a hyperparameter to be specified.

Given a network view $v \in \mathcal{V}$, we generate random walk pairs as in existing work [19, 18, 102]. Specifically, each random walk is of length $L \in \mathbb{N}$, and $\lfloor N/n^{(v)} \rfloor$ or $\lceil N/n^{(v)} \rceil$ random walks are sampled from each non-isolated node in view v, yielding a total of N random walks. For each node along any random walk, this node and any other node within a window of size $B \in \mathbb{N}$ constitute a random walk pair that is then added to $\mathcal{W}^{(v)}$.

Finally, we summarize both the MVN2VEC-CON algorithm and the MVN2VEC-REG algorithm in Algorithm 3.1.

3.5 EXPERIMENTS

In this section, we further corroborate the intuition of *preservation* and *collaboration*, and demonstrate the feasibility of simultaneously model these two characteristics. We first perform a case study on a series of synthetic multi-view networks that have varied extent of *preservation* and *collaboration*. Next, we introduce the real-world datasets, baselines, and experiment setting for more comprehensive quantitative evaluations. Lastly, we analyze the evaluation results and provide further discussion.

3.5.1 Case Study – Varied *preservation* and *collaboration* on Synthetic Data

In order to directly study the relative performance of different models on networks with varied extent of *preservation* and *collaboration*, we design a series of synthetic multi-view networks and experiment on a multi-class classification task.

We denote each of these synthetic networks by $S(p)$, where $p \in [0, 0.5]$ is referred to as intrusion probability. Each $S(p)$ has $4,000$ nodes and 2 views – v_1 and v_2. Furthermore, each node is associated to one of the 4 class labels – A, B, C, or D – and each class has

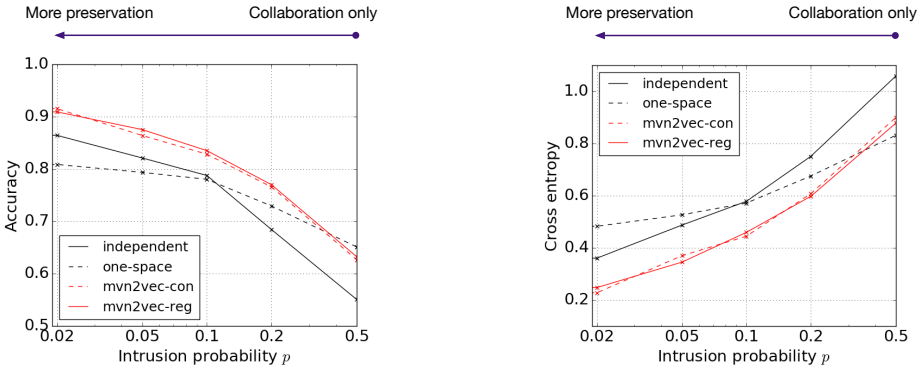


Figure 3.3: Classification results under accuracy and cross entropy on synthetic networks $S(p)$ with varied intrusion probability p, corresponding to different extent of *preservation* and *collaboration*.

exactly $1,000$ nodes. We first describe the process for generating $S(0)$ before introducing the more general $S(p)$ as follows:

1. Generate one random network over all nodes with label A or B, and another over all nodes with label C or D. Put all edges in these two random networks into view v_1.

2. Generate one random network over all nodes with label A or C, and another over all nodes with label B or D. Put all edges in these two random networks into view v_2.

To generate each of the four aforementioned random networks, we adopt the preferential attachment process with $2,000$ nodes and 1 edge to attach from a new node to existing nodes, where the preferential attachment process is a widely used method for generating networks with power-law degree distribution.

With this design for $S(0)$, view v_1 carries the information that nodes labeled A or B should be classified differently from nodes labeled C or D, while v_2 reflects that nodes labeled A or C are different from nodes labeled B or D. More generally, $S(p)$ are generated with the following tweak from $S(0)$: when putting an edge into one of the two views, with probability p, the edge is put into the other view instead of the view specified in the $S(0)$ generation process.

It is worth noting that larger p favors more *collaboration*, while smaller p favors more *preservation*. In the extreme case where $p = 0.5$, only *collaboration* is needed in the network embedding process. This is because every edge has equal probability to fall into view v_1 or view v_2 of $S(0.5)$, and there is hence no information carried specifically by either view that should be preserved.

On each $S(p)$, *independent*, *one-space*, MVN2VEC-CON, and MVN2VEC-REG are tested. On top of the embedding learned by each model, we apply logistic regression with cross entropy

Table 3.3: Basic statistics for the three real-world multi-view networks, where the number of edges specifies the total edge number from all network views.

Dataset	# views	# nodes	# edges
Snapchat	3	7,406,859	131,729,903
YouTube	3	14,900	7,977,881
Twitter	2	116,408	183,341

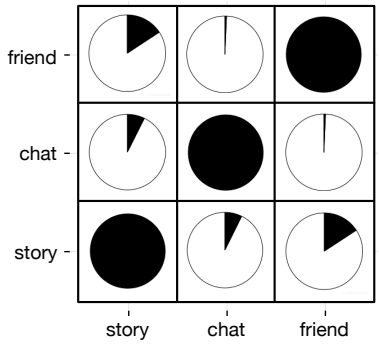


Figure 3.4: Agreement between each pair of views for Snapchat.

to carry out the multi-class evaluation tasks. All model parameters are tuned to the best for each model on a validation dataset sampled from the $4,000$ class labels. Classification accuracy and cross-entropy on a different test dataset are reported in Figure 3.3.

From Figure 3.3, we make three observations. (i) *independent* performs better than *one-space* in case p is small – when *preservation* is the dominating characteristic in the network – and *one-space* performs better than *independent* in case p is large – when *collaboration* is dominating. (ii) The two proposed MVN2VEC models perform better than both *independent* and *one-space* except when p is close to 0.5, which implies it is indeed feasible for MVN2VEC to achieve better performance by simultaneously model the two characteristics *preservation* and *collaboration*. (iii) When p is close to 0.5, *one-space* performs the best. This is expected because no *preservation* is needed in $S(0.5)$, and any attempts to additionally model *preservation* shall not boost, if not impair, the performance.

3.5.2 Data Description and Evaluation Tasks

We perform quantitative evaluations on three real-world multi-view networks: Snapchat, YouTube, and Twitter. The key statistics are summarized in Table 3.3, and we describe these datasets as follows.

Snapchat. Snapchat is a multimedia social networking service. On the Snapchat multi-view social network, each node is a user, and the three views correspond to friendship, chatting, and story viewing[1]. We perform experiments on the sub-network consisting of all users from Los Angeles. The data used to construct the network are collected from two consecutive weeks in the Spring of 2017. Additional data for downstream evaluation

[1]https://support.snapchat.com/en-US/a/view-stories

tasks are collected from the following week – henceforth referred to as week 3. We perform a multi-label classification task and a link prediction task on top of the user embedding learned from each network. For classification, we classify whether or not a user views each of the 10 most popular discover channels[2] according to the user viewing history in week 3. For each channel, the users who view this channel are labeled positive, and we randomly select 5 times as many users who do not view this channel as negative examples. These records are then randomly split into training, validation, and test sets. This is a multi-label classification problem that aims at inferring users' preference on different discover channels and can therefore guide product design in content serving. For link prediction, we predict whether two users would view the stories posted by each other in week 3. Negative examples are the users who are friends, but do not have story viewing in the same week. It is worth noting that this definition yields more positive examples than negative examples, which is the cause of a relatively high AUPRC score observed in experiments. These records are then randomly split into training, validation, and test sets with the constraint that a user appears as the viewer of a record in at most one of the three sets. This task aims to estimate the likelihood of story viewing between friends, so that the application can rank stories accordingly.

We also provide the Jaccard coefficient–based measurement on Snapchat in Figure 3.4. It can be seen that the cross-view agreement between each pair of views in the Snapchat network falls in between YouTube and Twitter presented in Section 3.2.

YouTube. YouTube is a video-sharing website. We use a dataset made publicly available by the Social Computing Data Repository [105][3]. From this dataset, a network with three views is constructed, where each node is a core user and the edges in the three views represent the number of common friends, the number of common subscribers, and the number of common favorite videos, respectively. Note that the core users are those from which the author of the dataset crawled the data, and their friends can fall out of the scope of the set of core users. Without user label available for classification, we perform only link prediction task on top of the user embedding. This task aims at inferring whether two core users are friends, which has also been used for evaluation by existing research [50]. Each core user forms positive pairs with his or her core friends, and we randomly select 5 times as many non-friend core users to form negative examples. Records are split into training, validation, and test sets as in the link prediction task on Snapchat.

Twitter. Twitter is an online news and social networking service. We use a dataset made

[2]https://support.snapchat.com/en-US/a/discover-how-to
[3]http://socialcomputing.asu.edu/datasets/YouTube

publicly available by the Social Computing Data Repository [106]. From this dataset, a network with two views is constructed, where each node is a user and the edges in the two views represent the number of replies and the number of mentions, respectively. Again, we evaluate by a link prediction task that infers whether two users are friends as in existing research [50]. The same negative example generation method and training–validation–test split method are used as in the YouTube dataset.

For each evaluation task on all three networks, training, validation, and test sets are derived in a shuffle split manner with a 80%–10%–10% ratio. The shuffle split is conducted for 20 times, so that mean and its standard error under each metric can be calculated. Furthermore, a node is excluded from evaluation if it is isolated from other nodes in at least one of the multiple views.

3.5.3 Baselines and Experimental Setup

In this section, we describe the baselines used to validate the utility of modeling *preservation* and *collaboration*, and the experimental setup for both embedding learning and downstream evaluation tasks.

Baselines. Quantitative evaluation results are obtained by applying downstream learner upon embedding learned by a given embedding method. Therefore, for fair comparisons, we use the same downstream learner in the same evaluation task. Moreover, since our study aims at understanding the characteristics of multi-view network embedding, we build all compared embedding methods from the same random work plus skip-gram approach with the same model inference method, as discussed in Section 3.4. Specifically, we describe the baseline embedding methods as follows:

- **Independent**. As briefly discussed in Section 3.3, the *independent* model first embeds each network view independently, and then concatenate them to find the final embedding $\mathbf{f}_u = \bigoplus_{v \in \mathcal{V}} \mathbf{f}_u^v \in \mathbb{R}^D$. This method is equivalent to MVN2VEC-CON when $\theta = 0$, and to MVN2VEC-REG when $\gamma = 0$. It preserves the information embodied in each view, but do not allow collaboration across different views in the embedding process.

- **One-space**. Also discussed in Section 3.3, the *one-space* model assumes the embedding of the same node to share model parameters across different views $\mathbf{f}_u = \mathbf{f}_u^v \in \mathbb{R}^D,\ \forall v \in \mathcal{V}$. It uses the same strategy to combine random walks generated from different views as with the proposed MVN2VEC methods. *one-space* enables different views to collaborate in learning a unified embedding, but do not preserve information specifically carried by each view.

- **View-merging**. The *view-merging* model first merges all network views into one unified view, and then learn the embedding of this single unified view. In order to comply with the assumed equal importance of different network views, we scale the weights of edges proportionally in each view, so that the total edge weights from all views are the same in the merged network. This method serves as an alternate approach to *one-space* in modeling *collaboration*. The difference between *view-merging* and *one-space* essentially lies in whether or not random walks can cross different views. We note that just like *one-space*, *view-merging* does not model *preservation*.

- **Single-view**. For each network view, the *single-view* model learns embedding from only this view, and neglects all other views. This baseline is used to verify whether introducing more than one view does bring in informative signals in each evaluation task.

Downstream learners. For fair comparisons, we apply the same downstream learner onto the features derived from each embedding method. Specifically, we use the scikit-learn[4] implementation of logistic regression with l-2 regularization and the SAG solver for both classification and link prediction tasks. For each task and each embedding method, we tune the regularization coefficient in the logistic regression to the best on the validation set. Following existing research [20], each embedding vector is normalized onto the unit l-2 sphere before feeding into downstream learners. In multi-label classification tasks, the features fed into the downstream learner is simply the embedding of each node, and we train an independent logistic regression model for each label. In link prediction tasks, features of node pairs are needed, and we derive such features by the Hadamard product of the two involved node embedding vectors as suggested by previous work [18].

Hyperparamters. For *independent*, MVN2VEC-CON, and MVN2VEC-REG, we set embedding space dimension $D = 128 \cdot |\mathcal{V}|$. For *one-space* and *view-merging*, we experiment with both $D = 128$ and $D = 128 \cdot |\mathcal{V}|$, and always report the better result between the two settings. For *single-view*, we set $D = 128$. To generate random walk pairs, we always set $L = 20$ and $B = 3$. For the Snapchat-LA network, we set $M = 10$ due to its large scale, and set $M = 50$ for all other datasets. The negative sampling rate K is set to be 5 for all models, and each model is trained for 1 epoch. In Figure 3.3, Table 3.4, Table 3.5, and Figure 3.6, θ and γ in the MVN2VEC models are also tuned to the best on the validation dataset. The impact of θ and γ on model performance is further presented and discussed in Section 3.5.5.

Metrics. For link prediction tasks, we use two widely used metrics: the area under the

[4]http://scikit-learn.org/stable/

Table 3.4: Mean of link prediction results with standard error in brackets.

Dataset	Metric	*single-view* (worst view)	*single-view* (best view)	*independent*	*one-space*	*view-merging*	MVN2VEC-CON	MVN2VEC-REG
Snapchat	ROC-AUC	0.587 (0.001)	0.592 (0.001)	0.617 (0.001)	0.603 (0.001)	0.611 (0.001)	0.626 (0.001)	**0.638** (0.001)
	AUPRC	0.675 (0.001)	0.677 (0.002)	0.700 (0.001)	0.688 (0.002)	0.693 (0.002)	0.709 (0.001)	**0.712** (0.002)
YouTube	ROC-AUC	0.831 (0.002)	0.904 (0.002)	**0.931** (0.001)	0.914 (0.001)	0.912 (0.001)	**0.932** (0.001)	**0.934** (0.001)
	AUPRC	0.515 (0.004)	0.678 (0.004)	**0.745** (0.003)	0.702 (0.004)	0.699 (0.004)	**0.746** (0.003)	**0.754** (0.003)
Twitter	ROC-AUC	0.597 (0.001)	0.715 (0.001)	0.724 (0.001)	0.737 (0.001)	0.741 (0.001)	0.727 (0.000)	**0.754** (0.001)
	AUPRC	0.296 (0.001)	0.428 (0.001)	0.447 (0.001)	0.466 (0.001)	0.469 (0.001)	0.453 (0.001)	**0.478** (0.001)

Table 3.5: Mean of classification results with standard error in brackets on the Snapchat multi-view network.

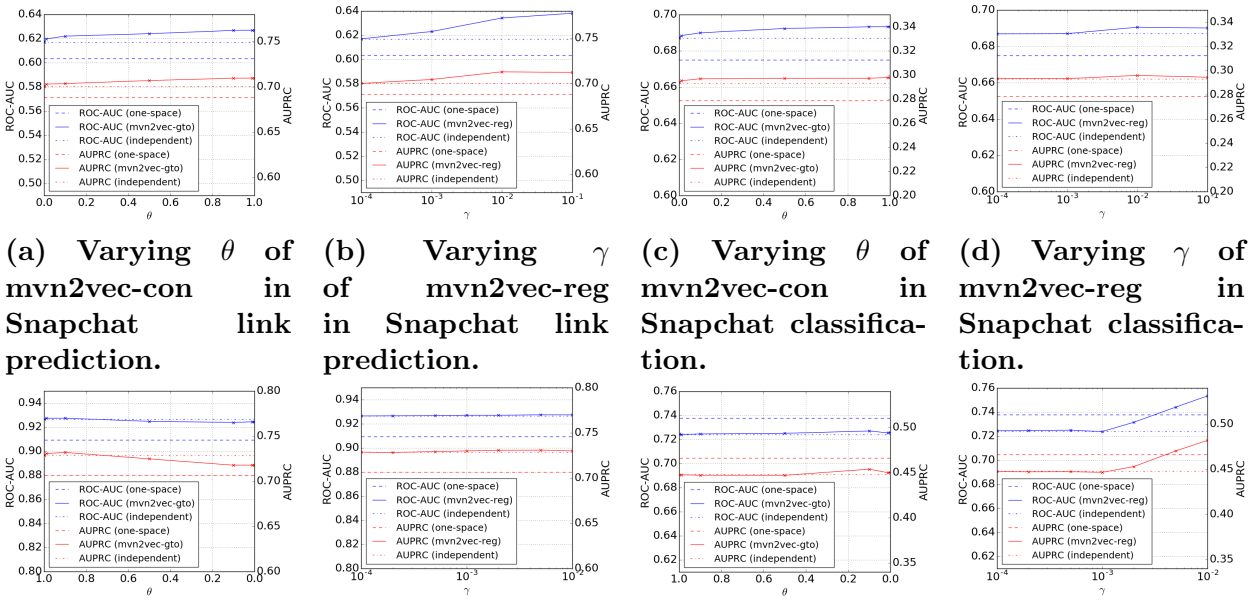
Dataset	Metric	*single-view* (worst view)	*single-view* (best view)	*independent*	*one-space*	*view-merging*	MVN2VEC-CON	MVN2VEC-REG
Snapchat	ROC-AUC	0.634 (0.001)	0.667 (0.002)	0.687 (0.001)	0.675 (0.001)	0.672 (0.001)	**0.693** (0.001)	0.690 (0.001)
	AUPRC	0.252 (0.001)	0.274 (0.002)	0.293 (0.002)	0.278 (0.001)	0.279 (0.001)	**0.298** (0.001)	0.296 (0.002)

receiver operating characteristic curve (ROC-AUC) and the area under the precision-recall curve (AUPRC). The receiver operating characteristic curve (ROC) is derived from plotting true positive rate against false positive rate as the threshold varies, and the precision-recall curve (PRC) is created by plotting precision against recall as the threshold varies. Higher values are preferable for both metrics. For multi-label classification tasks, we also compute the ROC-AUC and the AUPRC for each label, and report the mean value averaged across all labels.

3.5.4 Quantitative Evaluation Results on Real-World Datasets

The link prediction experiment results on three networks are presented in Table 3.4. For each dataset, all methods leveraging multiple views outperformed those using only one view, which justifies the necessity of using multi-view networks. Moreover, *one-space* and *view-merging* had comparable performance on each dataset. This is an expected outcome because they both only model *collaboration* and differ from each other merely in whether random walks are performed across network views.

On YouTube, the proposed MVN2VEC models perform as good but do not significantly exceed the baseline *independent* model. Recall that the need for *preservation* in the YouTube network is overwhelmingly dominating as discussed in Section 3.3. As a result, it is not surprising to see that additionally modeling *collaboration* does not bring about significant performance boost in such extreme case. On Twitter, *collaboration* plays a more important role than *preservation*, as confirmed by the better performance of *one-space* than *independent*. Furthermore, MVN2VEC-REG achieved better performance than all baselines, while MVN2VEC-CON outperformed *independent* by further modeling *collaboration*, but failed to



(a) Varying θ of mvn2vec-con in Snapchat link prediction.

(b) Varying γ of mvn2vec-reg in Snapchat link prediction.

(c) Varying θ of mvn2vec-con in Snapchat classification.

(d) Varying γ of mvn2vec-reg in Snapchat classification.

(e) Varying θ of mvn2vec-con in YouTube link prediction.

(f) Varying γ of mvn2vec-reg in YouTube link prediction.

(g) Varying θ of mvn2vec-con in Twitter link prediction.

(h) Varying γ of mvn2vec-reg in Twitter link prediction.

Figure 3.5: Performance of the mvn2vec models under varying hyperparameters regarding *preservation* and *collaboration*.

exceed *one-space*. This phenomenon can be explained by the fact that $\{\mathbf{f}_u^v\}_{v\in\mathcal{V}}$ in MVN2VEC-CON are set to be independent regardless of its hyperparameter $\theta \in [0, 1]$, and MVN2VEC-CON's capability of modeling *collaboration* is bounded by this design.

The Snapchat network used in our experiments lies in between YouTube and Twitter in terms of the need for *preservation* and *collaboration*. The proposed two MVN2VEC models both outperformed all baselines under all metrics. In other words, this experiment result shows the feasibility of gaining performance boost by simultaneously model *preservation* and *collaboration* without over-complicating the model or adding supervision.

The multi-label classification results on Snapchat are presented in Table 3.4. As with the previous link prediction results, the two MVN2VEC model both outperformed all baselines under all metrics, with a difference that MVN2VEC-CON performed better in this classification task, while MVN2VEC-REG outperformed better in the previous link prediction task. Overall, while MVN2VEC-CON and MVN2VEC-REG may have different advantages in different tasks, they both outperformed all baselines by simultaneously modeling *preservation* and *collaboration* on the Snapchat network, where both *preservation* and *collaboration* co-exist.

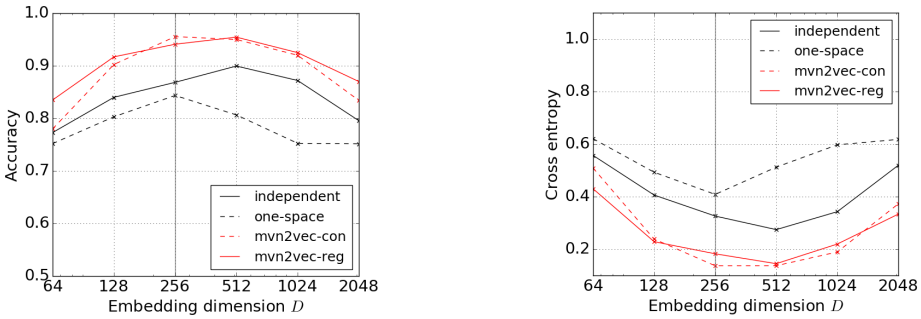


Figure 3.6: Classification results under accuracy and cross entropy on network $S(0)$ with varied embedding dimension D.

3.5.5 Hyperparameter Study

Impact of θ for mvn2vec-con and γ for mvn2vec-reg. With results presented in Figure 3.5, we first focus on the Snapchat network. Starting from $\gamma = 0$, where only *preservation* was modeled, MVN2VEC-REG performed progressively better as more *collaboration* kicked in by increasing γ. The peak performance was reached between 0.01 and 0.1. On the other hand, the performance of MVN2VEC-CON improved as θ grew. Recall that even in case $\theta = 1$, MVN2VEC-CON still have $\{\mathbf{f}_u^v\}_{v\in\mathcal{V}}$ independent in each view. This prevented MVN2VEC-CON from promoting more *collaboration*.

On YouTube, the MVN2VEC models did not significantly outperform *independent* no matter how θ and γ varied due to the dominant need for *preservation* as discussed in Section 3.3 and 3.5.4.

On Twitter, MVN2VEC-REG outperformed *one-space* when γ was large, while MVN2VEC-CON could not beat *one-space* for reason discussed in Section 3.5.4. This also echoed MVN2VEC-CON's performance on Snapchat as discussed in the first paragraph of this section.

Impact of embedding dimension. To rule out the possibility that *one-space* could actually preserve the view-specific information as long as the embedding dimension were set to be large enough, we further carry out the multi-class classification task on $S(0)$ under varied embedding dimensions. Note that $S(0)$ is used in this experiment because it has the need for modeling *preservation* as discussed in Section 3.5.1. As presented in Figure 3.6, *one-space* achieves its best performance at $D = 256$, which is worse than *independent* at $D = 256$, let alone the best performance of *independent* at $D = 512$. Therefore, one cannot expect *one-space* to preserve the information carried by different views by employing embedding space with large enough dimension.

Besides, all four models achieve their best performance with D in the vicinity of 256$\sim$512. Particularly, *one-space* requires the smallest embedding dimension to reach peak perfor-

mance. This is expected because, unlike the other models, *one-space* does not segment its embedding space to suit multiple views, and hence has more freedom in exploiting an embedding space with given dimension.

3.6 SUMMARY

We studied the characteristics that are specific and important to multi-view network embedding. *preservation* and *collaboration* were identified as two such characteristics in our practice of embedding real-world multi-view networks. We then explored the feasibility of achieving better embedding results by simultaneously modeling *preservation* and *collaboration*, and proposed two multi-view network embedding methods to achieve this objective. Experiments with various downstream evaluation tasks were conducted on a series of synthetic networks and three real-world multi-view networks with distinct sources, including two public datasets and an internal Snapchat dataset. Experiment results corroborated the presence and importance of *preservation* and *collaboration*, and demonstrated the effectiveness of the proposed methods.

CHAPTER 4: HETEROGENEITY EXISTS IN ASSOCIATION STRENGTH WITHIN ONE NETWORK

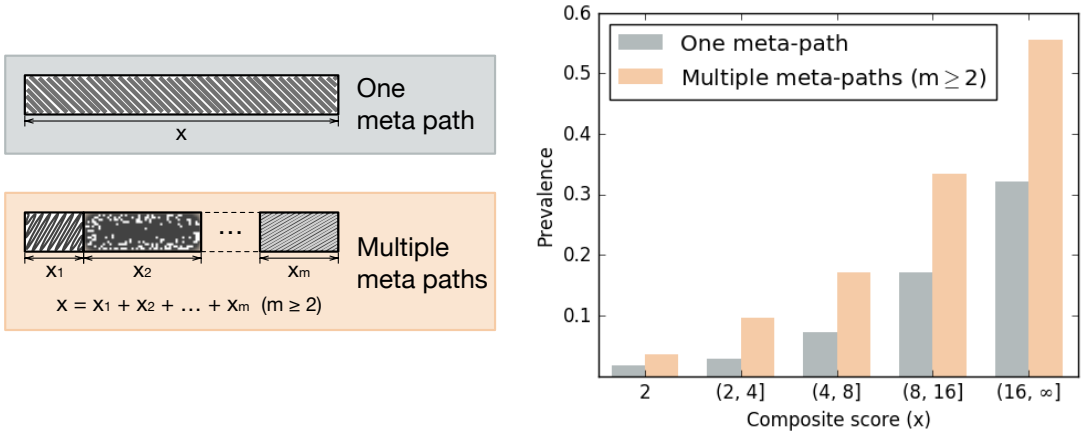
4.1 OVERVIEW

In real-world applications, objects of various types are often interconnected with each other. These objects, together with their relationship, form numerous heterogeneous information networks (HINs) [2, 14]. Bibliographical information network is a typical example, where researchers, papers, organizations, and publication venues are interrelated. A fundamental problem in HIN analysis is to define proper measures to characterize the relevance between node pairs in the network, which also benefits various downstream applications, such as similarity search, recommendation, and community detection [2, 14].

Most existing studies derive their HIN relevance measures on the basis of *meta-path* [2, 14, 22], which is defined as a concatenation of multiple node types linked by corresponding edge types. Based on the concept of *meta-path*, researchers have proposed PathCount, PathSim [22], and path constrained random walk [23] to measure relevance between node pairs. On top of these studies, people have explored the ideas of incorporating richer information [24, 25] and more complex typed structures [26, 27, 28] to define more effective relevance scoring functions, or adding supervision to derive task-specific relevance measures [29, 30, 31].

The probabilistic perspective. While building upon this powerful *meta-path* paradigm, we aim to additionally understand and model relevance from the probabilistic point of view. In this regard, we establish a probabilistic interpretation of existing HIN relevance measures, which is achieved by modeling the generating process of all path instances in an HIN and deriving the relevance of a node pair from the likelihood of observing the path instances connecting them. Relevance and likelihood can be connected by this approach because only a small portion of node pairs in an HIN are actually relevant; and a proper generating process has low likelihood to generate the path instances between each of these relevant node pairs. We will detailedly discuss this probabilistic interpretation in Sec. 4.3. Moreover, as a starting point for studying HIN relevance from the probabilistic perspective, we focus the scope of our work on the basic *unsupervised* scenario. Meanwhile, we assume that the meta-paths of interest are already given. That is, we defer the study on the cases with label information and meta-path selection to future work.

In order to determine relevance between any pair of nodes, we have the key insight that a path-based HIN relevance should contain three characteristics – *node visibility*, *path selectivity*, and *cross-meta-path synergy* – which we describe in the following paragraphs.



(a) (b)

Figure 4.1: (a) The same composite score (x) may be aggregated from different number of meta-paths, where score is represented by the length of the rectangles and each fill pattern represents a meta-path. (b) An observation made from an entity resolution task on the DBLP dataset that if linear combination is used to compute the composite score, node pairs with paths under multiple meta-paths are more likely to be relevant than those under only one meta-path. Prevalence is defined as the number of relevant node pairs divided by the total number of node pairs.

Node visibility. One straightforward way to derive relevance in an HIN is PathCount [22]. For a meta-path $t \in \{1, \ldots, T\}$, PathCount is defined as the number of paths, P_{st} or equivalently $P_{\langle uv \rangle t}$, under this meta-path between a node pair $s = (u, v) \in \mathcal{V} \times \mathcal{V}$, *i.e.*, $PathCount^{(t)}(u, v) \coloneqq P_{\langle uv \rangle t}$. One obvious drawback of this measure is that it favors nodes with high *node visibility*, *i.e.*, nodes with a large number of paths. To resolve this problem, [22] proposed to penalize PathCount by the arithmetic mean of the numbers of cycles attached to the two involved nodes, *i.e.*, $PathSim^{(t)}(u, v) \coloneqq \frac{2 \cdot P_{\langle uv \rangle t}}{P_{\langle uu \rangle t} + P_{\langle vv \rangle t}}$. A similar design to model *node visibility* can be found in JoinSim [107], which is defined as PathCount penalized by geometric mean of the cycle numbers.

Path selectivity. Given any method defining relevance score under one meta-path, a natural question is how to combine multiple meta-paths to derive a unified relevance score – henceforth referred to as the *composite score*. To achieve this goal, Sun et al. [22] proposed to assign different weights to different meta-paths, and compute the composite score via linear combination. Let $\mathbf{w} = \{w_1, \ldots, w_T\}$ with w_t being the weight for meta-path t, the composite score of PathCount is given by $PathCount_{\mathbf{w}}(u, v) \coloneqq \sum_{t=1}^{T} w_t \cdot PathCount^{(t)}(u, v)$. Similarly, one can define $PathSim_{\mathbf{w}}(u, v)$. This linear combination approach is adopted by follow-up works with multiple applications [2, 14], including personalized entity recommendation

problem [17], outlier detection [66, 16], *etc.* The weights assigned or inferred in these cases specify how selective each meta-path is. The larger the *path selectivity*, the more significant this meta-path is in contributing to the composite score.

Cross-meta-path synergy. Suppose linear combination is used to find the composite score as in the previous paragraph, the two scenarios shown in Fig. 4.1a would receive the same composite score (x), where x_i equals to the score from the i-th meta-path multiplied by the corresponding weight. However, we have the observation that, when meta-paths do not clearly correlate, the latter scenario tends to imply a higher relevance. We take an entity resolution task on the DBLP dataset as example, which aims to merge author mentions that refer to the same entity. In this task, each node stands for an author mention, and each meta-path represents that two author mentions have both published papers in one particular research area. We label two author mentions as relevant if and only if they refer to the same entity, and we use PathCount with uniform weights as an example to compute the composite score. Results presented in Fig. 4.1 shows that with the same composite score, node pairs associated by paths under multiple meta-paths are more likely to be relevant than those under only one meta-path. We refer to this phenomenon as *cross-meta-path synergy*. We interpret this phenomenon as given the occurrence of one path, the happenstance of another path under the same meta-path may not be surprising, while the co-occurrence of two paths under two uncorrelated meta-paths may be a strong signal of relevance. Moreover, we should also realize that not necessarily all meta-path pairs are uncorrelated, which has been observed in a special type of HIN [108]. This implies *cross-meta-path synergy* does not necessarily exist between all pairs of meta-paths, and we deem a good relevance measure should reflect this difference.

Challenges and contributions. Regarding the three pivotal characteristics for path-based HIN relevance discussed above, the major challenge lies in how to integrate all these characteristics in a unified framework. We tackle this challenge by studying path-based relevance from a probabilistic perspective, and deriving relevance measure from a generative model. Since the model parameters are trained to fit each HIN, the derived relevance measure enjoys the property of being data-driven. That is, the derived relevance measure is tailored for each HIN. Lastly, we summarize our contributions as follows:

1. We establish the probabilistic interpretation of existing path-based HIN relevance measures.

2. We identify and propose to model *cross-meta-path synergy*, an important characteristic in path-based HIN relevance.

3. We propose a novel relevance measure based on a generative model, which is data-driven and tailored for each HIN, and develop an inference algorithm with non-trivial tricks.

4. Experiments on two real-world HINs corroborate the effectiveness of our proposed model and relevance measure.

4.2 PRELIMINARIES

In this section, we introduce the concepts and notations used in our work.

Definition 4.1 (Heterogeneous Information Network)**.** *An **information network** is a directed graph $G = (\mathcal{V}, \mathcal{E})$ with a node type mapping $f : \mathcal{V} \to \mathcal{A}$ and an edge type mapping $g : \mathcal{E} \to \mathcal{R}$. Particularly, when the number of node types $|\mathcal{A}| > 1$ or the number of edge types $|\mathcal{R}| > 1$, the network is called a **heterogeneous information network** (HIN).*

Due to the typed essence of HINs, paths that associate node pairs can be grouped under different meta-paths. We formally define meta-paths as follows.

Definition 4.2 (Meta-Path)**.** *A **meta-path** is a concatenation of multiple nodes or node types linked by edge types.*

An example of a meta-path is [author] $\xrightarrow{\text{writes}}$ [paper] $\xrightarrow{\text{writes}^{-1}}$ [author], where a phrase in the brackets represents a node type and a phrase above the arrow refers to an edge type. When context is clear, we simply write [author]–[paper]–[author]. In our work, we study the relevance problem when a set of meta-paths of interest is predefined by users.

To ease presentation, we focus on unweighted HINs, and model path count defined as follows. Note that the path-based model to be proposed in our work can be extended to the weighted case.

Definition 4.3 (Path Count)**.** *The **path count** of a meta-path $t \in \{1, \ldots, T\}$ between a node pair $s = (u, v) \in \mathcal{V} \times \mathcal{V}$ is the number of concrete path instances under this meta-path that start from node u to node v, which is denoted by P_{st} or $P_{\langle uv \rangle t}$.*

Note that the relevance score given by the PathCount measure [22] is exactly the path count of a meta-path between a node pair.

Lastly, we introduce the probability distributions to be used.

Definition 4.4. *The probability density functions of three probability distributions used in our work are given as follows.*

1. *Exponential distribution* $\mathrm{Exp}\left(\tilde{\lambda}\right)$ *with rate parameter* $\tilde{\lambda} > 0$*:*

$$p(x) = \tilde{\lambda}\,\mathrm{e}^{\tilde{\lambda}x} \quad (x > 0).$$

2. *Gamma distribution* $\Gamma\left(\tilde{\alpha}, \tilde{\beta}\right)$ *with shape parameter* $\tilde{\alpha} > 0$ *and rate parameter* $\tilde{\beta} > 0$*:*

$$p(x) = \frac{\tilde{\beta}^{\tilde{\alpha}}}{\Gamma(\tilde{\alpha})} x^{\tilde{\alpha}-1}\,\mathrm{e}^{-\tilde{\beta}x} \quad (x > 0),$$

where $\Gamma(\tilde{\alpha}) = \int_0^\infty t^{\tilde{\alpha}-1}\,\mathrm{e}^{-t}\,dt$ *is the gamma function.*

3. *Symmetric Dirichlet distribution* $\mathrm{Dir}_L\left(\tilde{\alpha}\right)$ *of order* L *and concentration parameter* $\tilde{\alpha}$*:*

$$p(x_1, \ldots, x_L) = \frac{\Gamma\left(\tilde{\alpha}L\right)}{\Gamma\left(\tilde{\alpha}\right)^L} \prod_{i=1}^{L} x_i^{\tilde{\alpha}-1} \quad (x_i > 0 \text{ and } \sum_{i=1}^{L} x_i = 1),$$

where $\Gamma(\cdot)$ *is the gamma function.*

We denote $\mathrm{Exp}\left(x\,;\,\tilde{\lambda}\right) \coloneqq p(x)$ the probability density function of $\mathrm{Exp}\left(\tilde{\lambda}\right)$, and denote $x \sim \mathrm{Exp}\left(\tilde{\lambda}\right)$ if x is generated from $\mathrm{Exp}\left(\tilde{\lambda}\right)$. Similar notations are also used for $\Gamma\left(\tilde{\alpha}, \tilde{\beta}\right)$ and $\mathrm{Dir}_L\left(\tilde{\alpha}\right)$.

4.3 PROBABILISTIC INTERPRETATION OF EXISTING RELEVANCE MEASURES

In this section, we illustrate the probabilistic interpretation of existing path-based HIN relevance measures. We achieve this by studying the generating process of path counts between node pairs in an HIN, which contains a connection between relevance and the negative log likelihood. Suppose the path count under meta-path t between node pair s is generated from an exponential distribution

$$P_{st} \sim \mathrm{Exp}\left(\lambda\right),$$

with fixed rate λ, then in terms of the rank it yields, the negative log likelihood of all observed paths under meta-path t between node pair s will be equivalent to the PathCount under meta-path t

$$\begin{aligned} -LL^{(t)}(s) &= -\log(\lambda\,\mathrm{e}^{-\lambda P_{st}}) = \lambda P_{st} - \log\lambda \\ &\propto P_{st} + const = PathCount^{(t)}(s) + const. \end{aligned}$$

Table 4.1: Summary of symbols

Symbol	Definition
$\mathcal{V}$	The set of all nodes
$\mathcal{S}$	The set of all nontrivial node pairs
$T \in \mathbb{N}$	The number of meta-paths
$K \in \mathbb{N}$	The number of generating patterns
$\boldsymbol{P} \in \mathbb{R}^{\lvert\mathcal{S}\rvert \times T}$	The observed path counts between node pairs over each meta-path
$\boldsymbol{\eta} \in \mathbb{R}^{T}$	The *path selectivity*
$\boldsymbol{\tau} \in \mathbb{R}^{\lvert\mathcal{S}\rvert}$	The *node pair visibility*
$\boldsymbol{\rho} \in \mathbb{R}^{\lvert\mathcal{V}\rvert}$	The *node visibility*
$\boldsymbol{\Theta} \in \mathbb{R}^{\lvert\mathcal{S}\rvert \times K}$	The generating patterns over meta-paths
$\boldsymbol{\Phi} \in \mathbb{R}^{K \times T}$	The choices of generating patterns between node pairs
$\alpha \in \mathbb{R}_{+}$	The shape parameter of the gamma prior
$\beta \in (0, 1)$	The concentration parameter of the Dirichlet prior

Further, if we assume path instances under different meta-paths are generated from exponential distribution with meta-path-specific rates $\mathbf{w} = (w_1, w_2, \ldots, w_T)$, *i.e.*, $P_{st} \sim \mathrm{Exp}\,(w_t)$, then the negative log likelihood of all observed path counts will be equivalent to PathCount with weights $\mathbf{w}$ for linear combination

$$\begin{aligned} -LL(s) &= -\log(\prod_t w_t \,\mathrm{e}^{-w_t P_{st}}) = \sum_t w_t P_{st} - \sum_t \log w_t \\ &= \sum_t w_t P_{st} + const = PathCount_{\mathbf{w}}(s) + const. \end{aligned}$$

Moreover, if we assume each node pair s has pair-specific generating rate proportional to a parameter κ_s, *i.e.*, $P_{st} \sim \mathrm{Exp}\,(w_t/\kappa_s)$, then the negative log likelihood of observed path counts will be $-LL(s) = \sum_t w_t \cdot \frac{P_{st}}{\kappa_s} + T \log \kappa_s + const$. For node pair $s = (u, v)$, if we drop the logarithm term and set κ_s to be the arithmetic mean of the cycle count of the involved nodes u and v, the formula becomes

$$\sum_t w_t \cdot \frac{2 \cdot P_{\langle uv \rangle t}}{P_{\langle uu \rangle t} + P_{\langle vv \rangle t}} = PathSim_{\mathbf{w}}(s)$$

which is identical to PathSim with weights $\mathbf{w}$ for linear combination. In lieu of arithmetic mean, if we set κ_s to be the geometric mean of the same quantities, we get $\sum_t w_t \cdot \frac{P_{\langle uv \rangle t}}{\sqrt{P_{\langle uu \rangle t} \cdot P_{\langle vv \rangle t}}}$, which is identical to JoinSim with weights $\mathbf{w}$ for linear combination. Note that all the relevance measures discussed in this section are special cases of our relevance measure to be proposed in the next section.

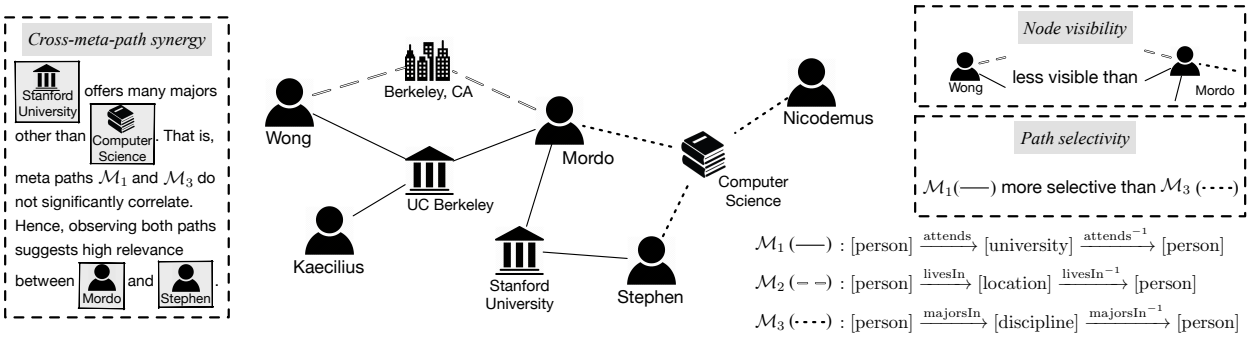


Figure 4.2: Toy example for one part of an HIN, consisting of four node types: person, university, location, and discipline.

4.4 PROPOSED MODEL AND RELEVANCE

With the relevance–likelihood connection established in Sec. 4.3, we propose our **P**ath-based **Re**levance from **P**robabilistic perspective (PReP) likewise by modeling the generating process of path counts between node pairs, and further aim to model the three important characteristics. In a nutshell, the proposed generative-model-based relevance measure consists of two major parts: (i) inferring model parameters by finding the maximum a posteriori (MAP) estimate to fit the input HIN, and (ii) deriving relevance score between any node pair based on the learned model.

4.4.1 The PReP Model

Following the existing HIN relevance measures discussed in Sec. 4.3, we assume the path count, P_{st} or $P_{\langle uv\rangle t}$, between node pair $s = (u, v)$ under meta-path t is generated from an exponential distribution with rate λ_{st}, *i.e.*, $P_{st} \sim \text{Exp}(\lambda_{st})$. To capture *node visibility*, *path selectivity*, and *cross-meta-path synergy*, we must design λ_{st} in a way that can model these three characteristics.

According to the property of exponential distribution, if a random variable X is generated from $\text{Exp}\left(\tilde{\lambda}\right)$, then the expectation of X will be $1/\tilde{\lambda}$. Bearing this in mind, we introduce three components to model the three characteristics as follows.

- Both the *node visibility* of u and that of v affect the generation of path instances. We consider the visibility of this pair of node as *node pair visibility*, τ_s, which is positively correlated with the expectation of P_{st}.

- We let path instances under the same meta-path share the same *path selectivity*. Denote η_t the *path selectivity* for meta-path t. η_t is negatively correlated with the expec-

tation of P_{st}.

- Each node pair with paths in between can be linked by path instances under a different set of meta-paths. We assume an underlying *meta-path distribution* $\boldsymbol{\psi}_s = [\psi_{s1}, \ldots, \psi_{sT}]$ for node pair s, where $\sum_{t=1}^{T} \psi_{st} = 1$ and $\psi_{st} \geq 0$. As a distribution over meta-paths, $\boldsymbol{\psi}_s$ models the semantics of the relevance between this node pair, because each meta-path carries its own semantic meaning. With further design to be introduced, $\boldsymbol{\psi}_s$ also serves as the basis to capture *cross-meta-path synergy*. ψ_{st} is positively correlated with the expectation of P_{st}.

Putting the above three components together considering their correlation with the expectation of P_{st}, we find path count generating process as

$$P_{st} \sim \text{Exp}\left(\frac{\eta_t}{\tau_s \psi_{st}}\right), \tag{4.1}$$

where the detailed illustration and design of the three components are to be further discussed in this section. Note that while we only discuss unweighted HINs in our work, the use of exponential distribution in Eq. (4.1) enables the model to handle weighted HINs, where paths are associated with real-valued path strengths, and P_{st} may not be integers to reflect the path strengths.

Since node pairs with no paths under any predefined meta-path should trivially receive the lowest possible relevance score, we only model the generation of path counts between node pairs with paths in between – henceforth referred to as nontrivial node pairs – and we denote $\mathcal{S}$ the set of all nontrivial node pairs.

Illustrative example. To better illustrate how each component design affects the path generation process, we present a toy example in Fig. 4.2, which shows a part of an HIN with four node types: person, university, location, and discipline. We concern three meta-paths in this network: $\mathcal{M}_1$: [person] $\xrightarrow{\text{attends}}$ [university] $\xrightarrow{\text{attends}^{-1}}$ [person], $\mathcal{M}_2$: [person] $\xrightarrow{\text{livesIn}}$ [location] $\xrightarrow{\text{livesIn}^{-1}}$ [person], $\mathcal{M}_3$: [person] $\xrightarrow{\text{majorsIn}}$ [discipline] $\xrightarrow{\text{majorsIn}^{-1}}$ [person].

Decoupling node pair visibility. To model *node visibility*, we decouple node pair visibility τ_s in Eq. (4.1) into two parts as in PathSim and JoinSim discussed in Sec. 4.3. The two parts correspond to the *node visibility* ρ_u and ρ_v, respectively, where $s = (u, v)$, and $\rho_z > 0$ for all $z \in \mathcal{V}$. In our design, we let

$$\tau_{(u,v)} = \rho_u \rho_v \tag{4.2}$$

as in JoinSim because decoupling by multiplication eases model inference, which will be made clear in the next paragraph.

Since a trivial rescaling – multiplying all ρ_z by a constant and multiplying all η_t by the square of the same constant – leads to exactly the same model (Eq. (4.1)), we further regularize ρ_z by a gamma prior with a constant rate parameter

$$\rho_z \sim \Gamma(\alpha, 1). \tag{4.3}$$

Note that we arbitrarily set the rate parameter to be 1 since the shape of the distribution is solely determined by the shape parameter α. We choose gamma distribution as the prior for ρ_z because it is the conjugate prior for the exponential distribution, and this fact will largely facilitate the inference algorithm as we will show in Sec. 4.5.2. To determine the shape parameter α, we fit the gamma distribution to the total path count each node has, $\{\sum_{t=1}^{T}\sum_{\tilde{z}\in\mathcal{V}} P_{\langle z\tilde{z}\rangle t}\}_{z\in\mathcal{V}}$, in the HIN as a rough prior information.

Path selectivity at meta-path level. We assume path instances under meta-path t share the same *path selectivity* η_t. In the scope of our work, where supervision is not available, we assume uninformative prior on η_t. In future work where supervision is provided, we can further learn η_t by minimizing the difference between supervision and model output to derive a task-specific relevance measure.

Cross-meta-path synergy and generating patterns. As discussed in Chapter 1, we have observed the existence of *cross-meta-path synergy* in real-world HIN, and this characteristic has not been modeled by existing HIN relevance measures. In case meta-paths do not correlate, we may simply add a Dirichlet prior, with concentration parameter smaller than 1, over meta-path distribution $\boldsymbol{\psi}_s$ for all node pair s. This use of Dirichlet prior resembles latent Dirichlet allocation (LDA) [109], where the Dirichlet prior prefers sparse distributions, *i.e.*, most entries of $\boldsymbol{\psi}_s$ tend to be 0. Therefore, the co-occurrence of paths under different meta-paths gets a lower likelihood from this prior, and attains a higher relevance score under our relevance–likelihood connection.

However, in reality, it would not be surprising to see two people attending UC Berkeley also both live in the City of Berkeley. This implies *cross-meta-path synergy* does not necessarily exist between all pairs of meta-paths, *e.g.*, it may not exist between meta-path $\mathcal{M}_1$ and meta-path $\mathcal{M}_2$ in the toy example of Fig. 4.2. To address this situation, we introduce a new component – *generating patterns*. Each of a total of K generating patterns is a distribution over the T meta-paths, where meta-paths that often co-occur between node pairs will also be included in a common generating pattern, and when a node pair s generates a path instance in between, it would first choose generating pattern k with probability φ_{sk}, and then choose meta-path t from this generating pattern with probability θ_{kt}. Formally, we describe this

Table 4.2: Existing measures cannot yield desired relevance, unless we assert $\mathcal{M}_3$ (discipline) is always more selective than $\mathcal{M}_2$ (location), while PReP can achieve this by recognizing the co-occurrence of multiple generating patterns.

Measure	Node Pair	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	Composite	Truth
PathCount	*Mordo & Wong*	1	1	0	$w_1 + w_2$	−
	Mordo & Stephen	1	0	1	$w_1 + w_3$	+
PathSim	*Mordo & Wong*	0.67	1	0	$0.67w_1 + w_2$	−
	Mordo & Stephen	0.67	0	1	$0.67w_1 + w_3$	+
RWR ($C = 0.9$)	*Mordo & Wong*	0.29	0.47	0	$0.29w_1 + 0.47w_2$	−
	Mordo & Stephen	0.25	0	0.31	$0.25w_1 + 0.31w_3$	+
PReP	*Mordo & Wong*	1 generating pattern				−
	Mordo & Stephen	2 generating patterns				+

process as

$$\psi_{st} = \sum_{k=1}^{K} \varphi_{sk}\theta_{kt}, \tag{4.4}$$

where $\boldsymbol{\varphi}_s = [\varphi_{s1}, \ldots, \varphi_{sK}]$ is node pair s's choices of generating patterns, such that $\sum_{k=1}^{K} \varphi_{sk} = 1$, $\varphi_{sk} \geq 0$; and $\boldsymbol{\theta}_k = [\theta_{k1}, \ldots, \theta_{kT}]$ is generating pattern k's distribution over meta-paths, such that $\sum_{t=1}^{T} \theta_{kt} = 1$, $\theta_{kt} \geq 0$.

A symmetric Dirichlet prior is then enforced on $\boldsymbol{\varphi}_s$, so that synergy will be recognized between and only between meta-paths from different generating patterns

$$\boldsymbol{\varphi}_s \sim \mathrm{Dir}_K(\beta), \tag{4.5}$$

where $\beta \in (0, 1)$ is the concentration hyperparameter.

With this design, our model gives a lower likelihood and higher relevance score to *Mordo* and *Stephen* (same university, same major) than *Mordo* and *Wong* (attending UC Berkeley and living in the City of Berkeley) in the toy example of Fig. 4.2 by learning a generating pattern that includes both $\mathcal{M}_1$ and $\mathcal{M}_2$. Whereas, other relevance measures cannot achieve this desired relationship as presented in Tab. 4.2, unless we set the weights $w_2 > w_3$, or equivalently assert $\mathcal{M}_2$ (location) is always less selective than $\mathcal{M}_3$ (discipline).

The unified model. For notation convenience, we use the bold italic form to represent the corresponding matrix or vector of each symbol with subscripts. For instance, the (s, t) element of $\boldsymbol{P}$ is P_{st} and the t-th element of $\boldsymbol{\eta}$ is η_t. Under this notation, combining Eq. (4.1), (4.3), and (4.5), with Eq. (4.2) and (4.4) substituted into Eq. (4.1), yields the total likelihood

of the full PReP model

$$\begin{aligned}\mathcal{L} &= p(\boldsymbol{P}, \boldsymbol{\eta}, \boldsymbol{\rho}, \boldsymbol{\Phi}, \boldsymbol{\Theta} \mid \alpha, \beta) \\ &= \left\{ \prod_{u \in \mathcal{V}} \Gamma\left(\rho_u \ ; \ (\alpha, 1)\right) \right\} \cdot \left\{ \prod_{s \in \mathcal{S}} \mathrm{Dir}_K\left(\boldsymbol{\varphi}_s \ ; \ \beta\right) \right\} \\ &\quad \cdot \left\{ \prod_{\substack{s \in \mathcal{S} \\ (u,v)=s}} \prod_{t=1}^{T} \mathrm{Exp}\left(P_{st} \ ; \ \frac{\eta_t}{\rho_u \rho_v \sum_{k=1}^{K} \varphi_{sk} \theta_{kt}} \right) \right\} \end{aligned} \tag{4.6}$$

4.4.2 The PReP Relevance Measure

Given the unified model (Eq. (4.6)), we have two options to derive relevance measure using likelihood: (i) find the maximum a posteriori estimate for all parameters and compute the total likelihood of the observed data, and (ii) consider all model parameters as hidden variables and define the relevance as the marginal likelihood of the observed data. However, the marginal likelihood does not have a closed-form representation in our case, nor can we approximate it with regular Markov chain Monte Carlo algorithms due to the large number of hidden variables. Therefore, we adopt the first option and defer the other to future work.

Once the model parameters $\{\boldsymbol{\eta}, \boldsymbol{\rho}, \boldsymbol{\Phi}, \boldsymbol{\Theta}\}$ are estimated, we define the PReP relevance for a node pair $s = (u, v)$ as the negative log-likelihood involving this node pair, $-\log p(\boldsymbol{P}_{s,:}, \boldsymbol{\varphi}_s \mid \boldsymbol{\Theta}, \boldsymbol{\rho}, \boldsymbol{\eta}, \alpha, \beta)$, without the log term as in the derivation of PathSim in Sec. 4.3

$$r(s) = \sum_{t=1}^{T} \frac{P_{st}}{\rho_u \rho_v \eta_t \sum_{k=1}^{K} \varphi_{sk} \theta_{kt}} + (1 - \beta) \sum_{k=1}^{K} \log \varphi_{sk}. \tag{4.7}$$

Note that PathCount, PathSim, and JoinSim discussed in Sec. 4.3 are special cases of this PReP relevance measure, when $\{\boldsymbol{\eta}, \boldsymbol{\rho}, \boldsymbol{\Phi}, \boldsymbol{\Theta}\}$ are heuristically specified accordingly.

4.5 MODEL INFERENCE

In this section, we introduce the inference algorithm for the PReP model (Eq. (4.6)) proposed in Sec. 4.4.

4.5.1 The Optimization Problem

We find the maximum a posteriori (MAP) estimate for model parameters by minimizing the negative log-likelihood of the proposed model (Eq. 4.6), which, with an offset of a constant, is given by

$$\begin{aligned}\mathcal{O} = &\sum_{u\in\mathcal{V}}(\rho_u - (\alpha-1)\log\rho_u) - (\beta-1)\sum_{s\in\mathcal{S}}\sum_{k=1}^{K}\log\varphi_{sk} \\ &+ T\sum_{(u,v)\in\mathcal{S}}(\log\rho_u + \log\rho_v) - |\mathcal{S}|\sum_{t=1}^{T}\log\eta_t \\ &+ \sum_{\substack{s\in\mathcal{S}\\(u,v)=s}}\sum_{t=1}^{T}\left[\log\sum_{k=1}^{K}\varphi_{sk}\theta_{kt} + \frac{\eta_t P_{st}}{\rho_u\rho_v\sum_{k=1}^{K}\varphi_{sk}\theta_{kt}}\right], \end{aligned} \tag{4.8}$$

and the optimization problem is therefore

$$\min_{\boldsymbol{\eta},\boldsymbol{\rho},\boldsymbol{\Phi},\boldsymbol{\Theta}} \mathcal{O}(\boldsymbol{\eta},\boldsymbol{\rho},\boldsymbol{\Phi},\boldsymbol{\Theta}). \tag{4.9}$$

We solve the above minimization problem with an iterative algorithm to be detailed in the following Sec. 4.5.2.

4.5.2 The Inference Algorithm

We iteratively update one of $\boldsymbol{\eta}$, $\boldsymbol{\rho}$, $\boldsymbol{\Phi}$, and $\boldsymbol{\Theta}$ when the others are fixed. The inference algorithm is summarized in Algorithm 4.1.

Update $\boldsymbol{\eta}$ given $\{\boldsymbol{\rho},\boldsymbol{\Phi},\boldsymbol{\Theta}\}$. Once given $\boldsymbol{\rho}$, $\boldsymbol{\Phi}$, and $\boldsymbol{\Theta}$, the optimal $\boldsymbol{\eta}$ that minimizes $\mathcal{O}$ in Eq. (4.8) has a closed-form solution. One can derive this closed-form update formula by looking back to the total likelihood Eq. (4.6), since

$$\begin{aligned}\mathcal{L} &\propto \prod_{s\in\mathcal{S}}\prod_{t=1}^{T}\mathrm{Exp}\left(P_{st}\ ;\ \frac{\eta_t}{\tau_s\sum_{k=1}^{K}\varphi_{sk}\theta_{kt}}\right) \\ &= \prod_{t=1}^{T}\left[\mathrm{Exp}\left(\frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\frac{P_{st}}{\tau_s\sum_{k=1}^{K}\varphi_{sk}\theta_{kt}}\ ;\ \eta_t\right)\right]^{|\mathcal{S}|},\end{aligned}$$

where $\tau_s = \rho_u\rho_v$ for node pair $s = (u,v)$. Using the property of exponential distributions,

Algorithm 4.1: Inference algorithm for the PReP model

Input : the observed path counts $\boldsymbol{P}$ and the hyperparameters
Output: the model parameters $\boldsymbol{\eta}$, $\boldsymbol{\rho}$, $\boldsymbol{\Phi}$, and $\boldsymbol{\Theta}$

```
begin
    Initialize ρ, Φ, and Θ
    while not converged do
        Update η by the closed-form Eq. (4.10)
        while not converged do
            for u ∈ V do
                Update ρ_u by the closed-form solution to Eq. (4.11)
        Update Φ via parallelized PGD with gradient in Eq. (4.13)
        Update Θ via PGD with gradient in Eq. (4.12)
```

we find the $\boldsymbol{\eta}$ that maximizes $\mathcal{L}$, and hence minimizes $\mathcal{O}$, can be computed by

$$\eta_t = \left(\frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \frac{P_{st}}{\tau_s \sum_{k=1}^{K} \varphi_{sk} \theta_{kt}} \right)^{-1}. \tag{4.10}$$

Update $\boldsymbol{\rho}$ given $\{\boldsymbol{\eta}, \boldsymbol{\Phi}, \boldsymbol{\Theta}\}$. Unlike $\boldsymbol{\eta}$, closed-form formula for updating $\boldsymbol{\rho}$ does not exist because (i) $\boldsymbol{\rho}$ has an informative prior, and (ii) the generating process for paths between node pair (u, v) involves the coupling of ρ_u and ρ_v. Fortunately, the gamma distribution is the conjugate prior to the exponential distribution. Therefore, for each u, when the rest $\{\rho_v\}_{v \neq u}$ are fixed, the closed-form update formula for ρ_u can be derived as follows. Denote $\{\xi_s\}_{s \in \mathcal{S}}$ the following quantities that are fixed during the $\boldsymbol{\rho}$ update phase

$$\xi_s \coloneqq \sum_{t=1}^{T} \frac{\eta_t P_{st}}{\sum_{k=1}^{K} \varphi_{sk} \theta_{kt}},$$

and we have $\frac{\partial \mathcal{O}}{\partial \rho_u} = \sum_{\substack{v \in \mathcal{V} \setminus \{u\} \\ s=(u,v)}} \left[\sum_{t=1}^{T} \frac{1}{\rho_u} - \frac{\xi_s}{\rho_u^2 \rho_v} \right] - \frac{\alpha - 1}{\rho_u} + 1$. Setting this partial derivative to 0 leads to

$$\rho_u^2 + \left[(|\mathcal{V}| - 1) \cdot T - (\alpha - 1) \right] \rho_u - \sum_{\substack{v \in \mathcal{V} \setminus \{u\} \\ s=(u,v)}} \frac{\xi_s}{\rho_v} = 0. \tag{4.11}$$

Note that Eq. (4.11) is a single-variable quadratic equation with one positive and one negative roots. Furthermore, $\mathcal{O}$ is convex *w.r.t.* ρ_u on the positive half-axis, and the positive root is a minimum of $\mathcal{O}$. Therefore, the optimal ρ_u that minimizes $\mathcal{O}$ is given by the positive root of the quadratic equation (Eq. (4.11)), which has closed-form solution. Holistically,

we update $\boldsymbol{\rho}$ by iterating through $u \in \mathcal{V}$ to update ρ_u with the aforementioned closed-form solution to Eq. (4.11).

Update $\boldsymbol{\Theta}$ given $\{\boldsymbol{\eta}, \boldsymbol{\rho}, \boldsymbol{\Phi}\}$. To update $\boldsymbol{\Theta}$, we use the projected gradient descent (PGD) algorithm [110]. The gradient is given by

$$\frac{\partial \mathcal{O}}{\partial \boldsymbol{\Theta}} = \boldsymbol{\Phi}^\top \left[\frac{1}{\boldsymbol{\Phi}\boldsymbol{\Theta}} - \frac{\boldsymbol{P}}{(\boldsymbol{\tau}(\boldsymbol{\eta}^{\circ -1})^\top) \circ (\boldsymbol{\Phi}\boldsymbol{\Theta})^{\circ 2}} \right], \tag{4.12}$$

where $[\cdot] \circ [\cdot]$, $\frac{[\cdot]}{[\cdot]}$, and $[\cdot]^{\circ [\cdot]}$ are element-wise multiplication, division, and power. Additional constraint fed into PGD is that each row of $\boldsymbol{\Theta}$ lies in the standard $(T-1)$-simplex, *i.e.*, $\sum_{t=1}^{T} \theta_{kt} = 1$ for all $k \in \{1, ..., K\}$ and $\theta_{kt} \geq 0$ for all $(k, t) \in \{1, ..., K\} \times \{1, ..., T\}$. Projection onto the standard simplex or the direct product of multiple standard simplices can be achieved efficiently using the method introduced in [111].

Update $\boldsymbol{\Phi}$ given $\{\boldsymbol{\eta}, \boldsymbol{\rho}, \boldsymbol{\Theta}\}$. Similarly, we use PGD to update $\boldsymbol{\Phi}$, where the gradient is given by

$$\frac{\partial \mathcal{O}}{\partial \boldsymbol{\Phi}} = \left[\frac{1}{\boldsymbol{\Phi}\boldsymbol{\Theta}} - \frac{\boldsymbol{P}}{(\boldsymbol{\tau}(\boldsymbol{\eta}^{\circ -1})^\top) \circ (\boldsymbol{\Phi}\boldsymbol{\Theta})^{\circ 2}} \right] \boldsymbol{\Theta}^\top - \frac{\beta - 1}{\boldsymbol{\Phi}}. \tag{4.13}$$

However, directly updating the entire $\boldsymbol{\Phi}$ using PGD can be problematic, because the row number of $\boldsymbol{\Phi}$ is the same as the number of nontrivial node pairs, $|\mathcal{S}|$, which can be significantly larger than that of $\boldsymbol{\Theta}$.

Fortunately, we can decompose the update scheme for $\boldsymbol{\Phi}$ by rows, because each row is independent from the others. Specifically, we update each row s using PGD in parallel, with gradient $\frac{\partial \mathcal{O}}{\partial \boldsymbol{\Phi}_{s,:}} = \left[\frac{1}{\boldsymbol{\Phi}_{s,:}\boldsymbol{\Theta}} - \frac{\boldsymbol{P}_{s,:}}{(\tau_s(\boldsymbol{\eta}^{\circ -1})^\top) \circ (\boldsymbol{\Phi}_{s,:}\boldsymbol{\Theta})^{\circ 2}} \right] \boldsymbol{\Theta}^\top - \frac{\beta - 1}{\boldsymbol{\Phi}_{s,:}}$, and constraints $\sum_{k=1}^{K} \varphi_{sk} = 1$ for all $s \in \mathcal{S}$ and $\varphi_{st} \geq 0$ for all $(s, k) \in \mathcal{S} \times \{1, ..., K\}$.

4.5.3 Implementation Details

For program reproducibility, we provide details in parameter initialization and computational singularity handling.

Since the inference algorithm starts with updating $\boldsymbol{\eta}$, no initialization for $\boldsymbol{\eta}$ is needed. $\boldsymbol{\rho}$ is initialized by drawing random samples from its prior distribution, $\Gamma(\alpha, 1)$, where α is estimated from data as discussed in Sec. 4.4. $\boldsymbol{\Phi}$ is initialized uniformly at random within the row-wise simplex constraint. For $\boldsymbol{\Theta}$, the first T rows of this $K \times T$ matrix are initialized to be an identity matrix, because many node pairs with paths in between involve only one meta-path, and we initialize the rest $K - T$ rows uniformly at random within the row-wise simplex constraint. This choice is out of the consideration that the PReP model is not

convex over all parameters.

Dirichlet distribution is defined over open sets with unbounded probability density function. As a result, when using MAP, certain components of $\mathbf{\Phi}$ can be inferred to approach the singularities along the boundary. Therefore, in practice, we let $\mathbf{\Phi}$ to be bounded away from the boundary with an infinitesimal quantity δ, *i.e.*, each of its entries must not only be positive, but also be greater or equal to δ. In this way, we keep the capability of Dirichlet distribution in modeling *cross-meta-path synergy*, while ensuring the model is computationally meaningful. In our experiment, we set $\delta = 10^{-50}$. With this constraint, the domain of definition for $\mathbf{\Phi}$ is no longer a standard simplex as discussed in [111]. For this reason, we provide the algorithm for efficient projection onto this shrunken simplex, $\{\mathbf{x} \in \mathbb{R}^K | x_i \geq \delta, \sum_{i=1}^K x_i = 1\}$, which is required by the inference algorithm.

We provide the algorithm for efficient projection onto the standard simplex shrunk by δ, $\{\mathbf{x} \in \mathbb{R}^K | x_i \geq \delta, \sum_{i=1}^K x_i = 1\}$, in Algorithm 4.2.

Algorithm 4.2: Efficient projection onto shrunk simplex

Input : the original vector $\mathbf{z} \in \mathbb{R}^K$ and the shrinking factor δ
Output: the projection $\mathbf{x} \in \mathbb{R}^K$

begin
  Sort $\mathbf{z}$ into $\mathbf{u}$: $u_1 \geq u_2 \geq \ldots \geq u_K$
  $\rho \leftarrow \max\{1 \leq j \leq K | u_j + \frac{1}{j}(1 - \delta K - \sum_{i=1}^j u_i) > 0\}$
  $\lambda \leftarrow \frac{1}{\rho}(1 - \delta K - \sum_{i=1}^\rho u_i)$
  $x_i \leftarrow \max\{z_i + \lambda, 0\} + \delta$

The validity of this algorithm can be established in a way similar to the proof of the algorithm for standard simplex [111].

Note that if one wishes to evade the point estimation of parameters in the PReP model, Eq. (4.6), and thereby avoid computational singularity, they can treat all model parameters as hidden variables and derive relevance from the marginal likelihood of the observed data as discussed in Sec. 4.4.2. The exploration of this direction requires novel method, such as a sampling algorithm design for our model, to efficiently calculate marginal likelihood, and we defer this to future work.

4.6 EXPERIMENTS

In this section, we quantitatively evaluate the proposed model on two publicly available real-world HINs: Facebook and DBLP. We first describe the datasets and the unsupervised

tasks used for evaluation. Baselines and model variations for comparison are then introduced. Afterward, we present experiment results together with discussions, which demonstrate the advantage of using probability as the backbone of relevance.

4.6.1 Data Description and Evaluation Tasks

In this section, we introduce the two publicly available real-world datasets and the evaluation tasks.

The Facebook dataset. This dataset [112] contains nodes of 11 types, including user, major, degree, school, hometown, surname, location, employer, work-location, work-project, and other. It consists of $5,621$ nodes and $98,023$ edges, among which $4,167$ nodes are of the user type. We aim to determine the relevance between users, using 10 meta-paths, each of the form [user]–[X]–[user], where X is any of the above 11 node types except for other.

To derive ground truth label between user pairs for evaluation, we use being friends on Facebook as a proxy for being relevant. This dataset is collected by recruiting participants to label their own Facebook friends It consists of 10 distinct ego networks, where an ego network consists of one ego user and all her friends together with edges attached to these users. We hence perform one sub-task for each ego network, where the compared measures are used to calculate the relevance between all pairs of non-ego users in this ego network.

We use two evaluation metrics widely adopted for tasks with multiple relevant instances: the area under the receiver operating characteristic curve (ROC-AUC) and the area under precision-recall curve (AUPRC). The receiver operating characteristic curve (ROC) is created by plotting true positive rate against false positive rate as the threshold varies, while the precision-recall curve (PRC) is drawn by plotting precision against recall as the threshold varies. Higher values are more preferred for both ROC-AUC and AUPRC. We further average each of the above metrics across ego networks with the following methods – uni.: averaging over all ego networks uniformly; rel.: weighting by the number of relevant pairs in each ego network; tot.: weighting by the total number of pairs in each ego network.

The DBLP dataset. This dataset is derived from the DBLP dataset processed by Tang et al. [113] containing computer science research papers together with author names and publication venue associated to each paper. It consists of $13,697$ nodes and $19,665$ edges, among which $1,546$ nodes are of the author type. Notably, in this dataset, the same author name associated with two papers may not necessarily be the same person. Based on this fact, we design an entity resolution task as follows. First, we use the labels made available by Tang et al. [113] to group all author name mentions corresponding to one person to

Table 4.3: Quantitative evaluation results on two real-world datasets using the proposed measure, PReP, and other measures.

Dataset	Metric		PathCount		PathSim		JoinSim		SimRank		PReP			
			Mean	SD	Mean	SD	Mean	SD	Mean	SD	No-NV	No-PS	No-CS	(full)
Facebook	ROC-AUC	uni.	0.8056	0.8598	0.8367	0.8586	0.8326	0.8547	0.7977	0.8303	0.8310	0.6702	0.8689	**0.8850**
		rel.	0.8612	0.8879	0.8578	0.8888	0.8556	0.8872	0.8076	0.8596	0.8556	0.6713	0.8880	**0.9133**
		tot.	0.8558	0.8849	0.8577	0.8866	0.8557	0.8851	0.8096	0.8594	0.8547	0.6773	0.8893	**0.9139**
	AUPRC	uni.	0.2456	0.2832	0.2370	0.2845	0.2340	0.2803	0.2055	0.2435	0.2183	0.1650	**0.3273**	**0.3269**
		rel.	0.2496	0.3048	0.2142	0.2873	0.2117	0.2837	0.1764	0.2408	0.2067	0.1283	0.3354	**0.3486**
		tot.	0.2107	0.2542	0.1841	0.2460	0.1821	0.2432	0.1523	0.2071	0.1760	0.1089	0.3010	**0.3080**
DBLP	MRR	uni./rel.	0.8091	0.8130	0.6922	0.7003	0.7454	0.7538	0.6636	0.6738	0.8223	0.8494	0.8365	**0.8517**
		tot.	0.7839	0.7871	0.6612	0.6731	0.7128	0.7244	0.6302	0.6357	0.8234	**0.8407**	0.8264	0.8391

define an author node. In this way, an author node is linked to multiple papers written by her. Then, for each author name, we split the author node with the most author name mentions into two nodes, and we define two nodes to be relevant if and only if they actually refer to the same person. Finally, we perform one sub-task for each author name, where the compared measures are used to calculate the relevance between all pairs of nodes with the same author name.

We use 14 meta-paths in this task, each of the form [author]–[paper]–[venue domain]–[paper]–[author], where a node of the venue domain type corresponds to one of the 14 computer science research areas. The definition of the 14 areas is derived from the Wikipedia page: List of computer science conferences[1]. Since only one relevant pair exists in each sub-task, the mean reciprocal rank (MRR) is used as the evaluation metric, where, for each sub-task, the reciprocal rank is the reciprocal of the rank of the relevant pair. Higher values indicate better results for MRR. We also average the above metrics across different sub-tasks using three methods: uni., rel., and tot. Note that uni. and rel. are equivalent in this entity resolution task because each sub-task has exactly one relevant pair.

4.6.2 Baselines and Variations

In this section, we describe the meta-path-based baseline methods and variations of the PReP model, which are used to compare with our proposed full PReP model. Existing meta-path-based unsupervised HIN measures define relevance computation method on each meta-path and then use linear combination to find the composite score. Therefore, each baseline consists of two parts: (i) the base measure that calculates the relevance score on one meta-path, and (ii) the weights assigned to different meta-paths used in the linear combination. The 4 base measures we used are:

- **PathCount** [22]. $PathCount_{\mathbf{w}}(s) \coloneqq \sum_t w_t P_{st}$.

[1]https://en.wikipedia.org/wiki/List_of_computer_science_conferences

- **PathSim** [22]. $PathSim_{\mathbf{w}}(s) \coloneqq \sum_t w_t \cdot \frac{2 \cdot P_{\langle uv \rangle t}}{P_{\langle uu \rangle t} + P_{\langle vv \rangle t}}$.
- **JoinSim** [107]. $JoinSim_{\mathbf{w}}(s) \coloneqq \sum_t w_t \cdot \frac{P_{\langle uv \rangle t}}{\sqrt{P_{\langle uu \rangle t} \cdot P_{\langle vv \rangle t}}}$.
- **SimRank**. We adopt SimRank [64] with meta-path constraints. Let $\boldsymbol{A}$ be a matrix, where A_{uv} is the number of paths under this meta-path between node pair (u, v) after column normalization. The SimRank score is then given by S_{uv}, where $\boldsymbol{S}$ is the solution to $\boldsymbol{S} = \max\{C \cdot (\boldsymbol{A}^\top \boldsymbol{S} \boldsymbol{A}), \boldsymbol{I}\}$, and C is the decay factor to be specified. Note that we use SimRank instead of random walk with restart because SimRank is a symmetric relevance measure.

Without any supervision available, we use 2 heuristics to determine the weights $\mathbf{w}$ for linear combination.

- **Mean**. Let w_t be the reciprocal of the mean of all scores computed using the corresponding base measure on meta-path t.
- **SD**. Let w_t be the reciprocal of the standard deviation of all scores computed using the corresponding base measure on meta-path t. Note that this heuristic normalized the original score in the way similar to z-score.

Combining the aforementioned 4 base measures and 2 heuristic for setting weights, we have 8 baselines in total.

Additionally, we also experiment with three variations of PReP, which are partial models with one of the three components knocked out from the full PReP model.

- No *node visibility* (No-NV): Set $\boldsymbol{\rho} = \mathbf{1}_{|\mathcal{V}|}$, and do not update $\boldsymbol{\rho}$ during model inference.
- No *path selectivity* (No-PS): Set $\boldsymbol{\eta} = \mathbf{1}_T$, and do not update $\boldsymbol{\eta}$ during model inference.
- No *cross-meta-path synergy* (No-CS): Set $\boldsymbol{\Phi} = \mathbf{1}_{|\mathcal{V}| \times K}/K$, $\boldsymbol{\Theta} = \mathbf{1}_{|\mathcal{V}| \times T}/T$, and do not update $\boldsymbol{\Phi}$ and $\boldsymbol{\Theta}$ during model inference.

Note that $\mathbf{1}_M$ stands for all one column vector of size M and $\mathbf{1}_{M \times N}$ denotes all one matrix of size $M \times N$.

4.6.3 Effectiveness and Discussion

In this section, we present the quantitative evaluation results on both the Facebook and the DBLP datasets. We tune the decay factor C in the baseline measure, SimRank, to have the

best performance with $C = 0.5$ for both SimRank-Mean and SimRank-SD on Facebook, and $C = 0.8$ for SimRank-Mean, $C = 0.7$ for SimRank-SD on DBLP. We set hyperparameters of PReP as $K = 15$ and $\beta = 10^{-4}$ for Facebook and $K = 14$ and $\beta = 10^{-2}$ for DBLP. The choice of hyperparameters will be further discussed in this section.

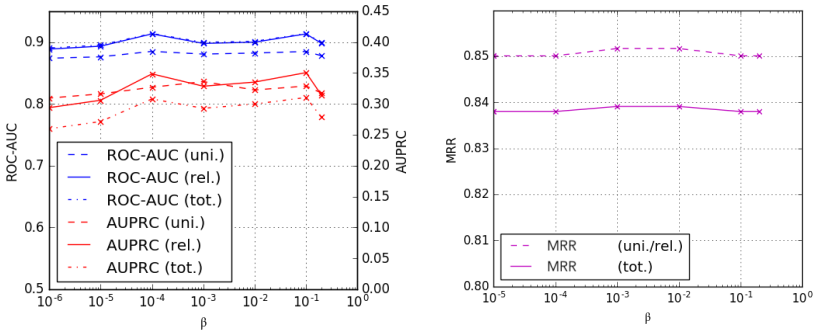
As presented in Tab. 4.3, PReP outperformed all 8 baselines under various metrics. Moreover, PReP outperformed its 3 variations under most metrics, suggesting each component of the model generally has a positive effect on the performance of the full PReP model. Note that under MRR (tot.), PReP performed slightly worse than PReP-No-PS, the partial model without η_t for *path selectivity*. This happened because, as discussed in Sec. 4.4, we cannot enforce task-specific design on *path selectivity* η_t due to the lack of supervision, and we expect *path selectivity* η_t to play a more important role in future work where relevance labels are provided as supervision.

Additionally, we have made the following observations.

Heuristic methods cannot yield robust relevance measures. Compared with PathCount, both PathSim and JoinSim further model *node visibility*, which penalizes the relevance with nodes that are highly visible. However, as Tab. 4.3 presents, PathSim and JoinSim cannot always outperform PathCount. Moreover, JoinSim performs better than PathSim on DBLP, while PathSim is slightly better than JoinSim on Facebook. We interpret these results as, PathSim and JoinSim model *node visibility* in a deterministic heuristic way. Unlike our generative-model-based measure that derives relevance measure based on parameters inferred from each HIN, the heuristic approaches adopted by PathSim and JoinSim have varying performance on different HINs. This suggests being data-driven is a favorable property of PReP.

Non-one-hot generating patterns help only when meta-paths correlate. In our experiment, we set $K = 14 = T$ for DBLP. Recall that we initialized the first T rows of $\mathbf{\Theta}$, the matrix representing the K generating patterns, to be T one-hot vectors corresponding to T meta-paths. We observed in the DBLP experiment that after model fitting, $\mathbf{\Theta}$ was still the same as its initialization, meaning each inferred generating pattern only generated path instances under exactly one meta-path. Moreover, by increasing the value of K, we did not see improvement in performance. This observation is inline with the situation that it is not frequently seen that two authors both publish papers in two distinct research areas, where the 14 areas on the Wikipedia page have been defined to be distinct areas including theory, software, parallel computing, *etc.* In this case, it is preferred to model synergy across every pair of meta-paths, and not to employ any non-one-hot generating patterns.

On the other hand, we used $K = 15 > T$ for Facebook, and we did observe non-one-hot



(a) Facebook **(b) DBLP**

Figure 4.3: Performance with varied β.

generating patterns after model fitting. The most popular non-one-hot generating pattern consisted of three meta-paths: [user]–[hometown]–[user], [user]–[school]–[user], and [user]–[user]–[user], where we define popularity of a generating pattern as the fraction of node pairs adopting this pattern, *i.e.*, $pop(k) = \sum_{s \in \mathcal{S}} \varphi_{sk}$. This generating pattern corresponds to two users sharing the same hometown, the same school, and having common friends. This scenario is common for two people sharing similar friend group back in the hometown school.

Sensitivity of β in modeling *cross-meta-path synergy*. In the PReP model (Eq. (4.6)) and relevance measure (Eq. (4.7)), the concentration parameter β of the Dirichlet prior controls the extent to which we boost *cross-meta-path synergy*. Experiment results in Fig 4.3 shows performance of PReP do not significantly change around the values we have set for β, *i.e.*, 10^{-4} for Facebook and 10^{-2} for DBLP.

4.7 RELATED WORK

In this section, we review the study on HIN relevance. The problem of deriving relevance between node pairs has been extensively studied for homogeneous information networks. Relevance measures of this type include the random walk based *Personalized PageRank* and *SimRank* [64], the neighbor-based *common neighbors* and *Jaccard's coefficient*, the path-based *Katz* [65], *etc.* To generalize relevance from the homogeneous networks to the typed heterogeneous case, researchers have been exploring from multiple perspectives. One perspective, as in *PathCount* and *PathSim* from [22] and *Path-Constrained Random Walk* from [23], is to first compute relevance score along each meta-path, and then glue scores from all types together via linear combination to establish the composite measure. A great many applications [66, 2, 14, 17, 16] based on this meta-path paradigm with linear combination have been proposed. Our proposed method follows this meta-path paradigm, but goes beyond

linear combination to model *cross-meta-path synergy* that we have observed from real-world HINs. Another perspective is to go beyond meta-path and derive relevance based on the more complex graph structures [26, 27]. While these approaches can yield good performance, they differ from our proposed methods for further entailing label information or expertise in designing graph structure. Also, they do not carry probabilistic interpretations. Besides, people have explored the idea of incorporating richer information [24, 25] to define more effective relevance scoring functions, or adding supervision to derive task-specific relevance measures [29, 30, 31]. While being valuable, these works are out of the scope of the problem we study in our work, where we address the basic, unsupervised case with no additional information as our starting point of studying HIN relevance from the probabilistic perspective.

4.8 SUMMARY

Inspired by the probabilistic interpretation of existing path-based relevance measures, we studied HIN relevance from a probabilistic perspective. We identified *cross-meta-path synergy* as one of the three characteristics that we deem important for HIN relevance. A generative model was proposed to derive a novel path-based relevance measure, PReP, which could capture the three important characteristics. An inference algorithm was also developed to find the maximum a posteriori (MAP) estimate of the model parameters, which entailed non-trivial tricks. Experiments on real-world HINs demonstrated the effectiveness of our relevance measure, which is data-driven and tailored for each HIN.

CHAPTER 5: HARNESSING HETEROGENEOUS ASSOCIATION BY IDENTIFYING ASPECTS OF AN HIN

5.1 OVERVIEW

In real-world applications, objects of different types interact with each other, forming heterogeneous relations. Such objects and relations, acting as strongly-typed nodes and edges, constitute numerous *heterogeneous information networks (HINs)* [2, 14]. HINs have received increasing interests in the past decade due to its capability of retaining the rich type information, as well as the accompanying wide applications such as recommender system [17], clustering [15], and outlier detection [16]. As an example, the IMDb network is an HIN containing information about users' preferences over movies and have five different node types: user, movie, actor, director, and genre.

Meanwhile, network embedding has recently emerged as a scalable unsupervised representation learning method [55, 18, 19, 40, 34, 20, 21]. In particular, network embedding learning projects the network into low-dimensional space, where each node is represented using a corresponding embedding vector and the relativity among nodes is preserved. With the semantic information transcribed from the networks, the embedding vectors can be directly used as node features in various downstream applications. We therefore use the two terms – the embedding of a node and the learned feature of a node – interchangeably in this chapter.

The heterogeneity in HINs poses a specific challenge for data mining and applied machine learning. We hence propose to study the problem of learning embedding in HINs with an emphasis on leveraging the rich and intrinsic type information. There are multiple attempts in studying HIN embedding or tackling specific application tasks using HIN embedding [32, 29, 33, 34]. Though these studies formulate the problem differently with respective optimization objectives, they share a similar underlining philosophy: using a unified objective function to embed all the nodes into *one* low-dimensional space.

Embedding all the nodes into *one* low-dimensional space, however, may lead to information loss. Take the IMDb network as example, where users review movies based on their preferences. Since each movie has several facets, users may review movies with emphasis over different facets. For instance, both Alice and Bob may like the movie Star Wars, but Alice likes it because of Carrie Fisher *(actor)*; while Bob likes it because it is a fantasy movie *(genre)*. Furthermore, suppose user Carlo likes both movies directed by Steven Spielberg and musicals. Due to the semantic dissimilarity between Steven Spielberg and musical, if this HIN were projected into one embedding space as visualized in the upper part of Figure 5.1,

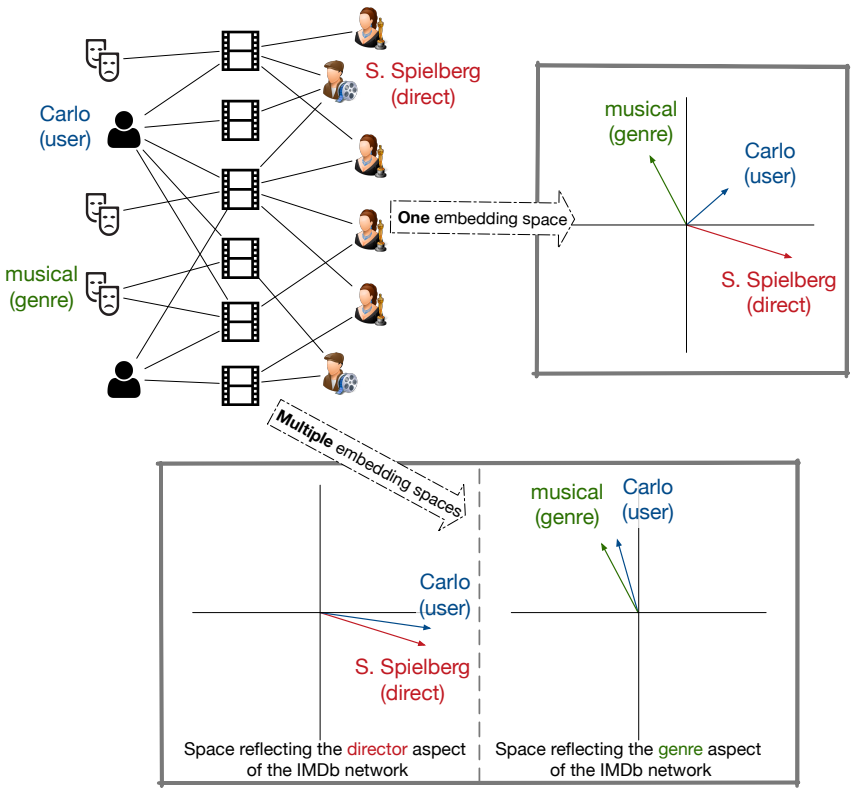


Figure 5.1: A toy example of node embeddings in an HIN. The upper left of the figure depicts the interactions among nodes, where users review movies and movies have various attributes. Carlo likes both musicals and movies directed by S. Spielberg. If all nodes were embedded to one space, Carlo would be close to neither musical nor S. Spielberg due to the dissimilarity between musical and S. Spielberg. However, by embedding the aspect related to director and that related to genre into separate spaces, Carlo could be close to S. Spielberg in one space, and close to musical in another.

musical *(genre)* and Steven Spielberg *(director)* would be distant from each other, while the user Carlo would be in the middle and close to neither of them. Therefore, it is of interest to obtain an embedding that can reflect Carlo's preference for both musicals and Spielberg's movies. To this end, we are motivated to embed the network into two distinct spaces: one for the aspect of genre information whereas the other for that of director information. In this case, Carlo could be close to musical *(genre)* in the first space and close to Steven Spielberg *(director)* in the second space as in the lower part of Figure 5.1.

In our work, we propose a flexible embedding learning framework – AspEm– for HINs that mitigates the incompatibility among aspects via considering each aspect separately. The use of aspects is motivated by the intuition that very distinct relationship could exist between components of a typed network, which has been observed in a special type of HIN [108].

Moreover, we demonstrate the feasibility of selecting a set of representative aspects for any HIN using statistics of the network without additional supervision.

It is worth noting that most existing embedding learning methodologies can be extended based on AspEm using the principle that different aspects should reside in different embedding spaces. Therefore, AspEm is a principled and flexible framework that has the potential of inheriting the merits of other embedding learning methods. To the best of our knowledge, this is the first work to study the property of multiple aspects in HIN embedding learning. Lastly, we summarize our contributions as follows:

1. We provide a key insight regarding incompatibility in HINs that each HIN can have multiple aspects that do not align with each other. We thereby identify that embedding algorithms employing only one embedding space may lose subtlety of the given HIN.

2. We propose a flexible HIN embedding framework, named AspEm, that can mitigate the incompatibility among multiple aspects via considering the semantic information regarding each aspect separately.

3. We propose an aspect selection method for AspEm, which demonstrates that a set of representative aspects can be selected from any HIN using statistics of the network without additional supervision.

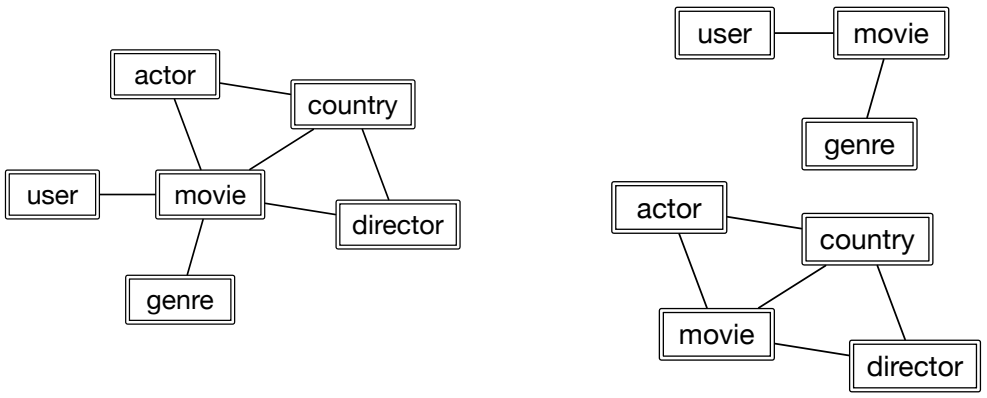
4. We conduct quantitative experiments on two real-world datasets with various evaluation tasks, which validate the effectiveness of the proposed framework.

5.2 PROBLEM DEFINITION

In this section, we formally define the problem of learning embedding from aspects of HINs and related notations.

Definition 5.1 (HIN)**.** *An **information network** is a directed graph $G = (\mathcal{V}, \mathcal{E})$ with a node type mapping $\varphi : \mathcal{V} \to \mathcal{T}$ and an edge type mapping $\psi : \mathcal{E} \to \mathcal{R}$. Particularly, when the number of node types $|\mathcal{T}| > 1$ or the number of edge types $|\mathcal{R}| > 1$, the network is called a **heterogeneous information network** (HIN) [14].*

In addition, when the network is weighted and directed, we use $W_{uv}^{(r)}$ to denote the weight of an edge $e \in \mathcal{E}$ with type $r \in \mathcal{R}$ that goes out from node u and into node v. $D_u^{O(r)}$ and $D_u^{I(r)}$ represent the outward degree of node u (*i.e.*, the sum of weights associated with all edges in type r going outward from u) and the inward degree of node u (*i.e.*, the sum of weights associated with all edges in type r going inward to u), respectively. For unweighted



(a) Schema **(b) Two aspects**

Figure 5.2: The schema and two aspects of an toy HIN with six node types: movie, director, actor, genre, country, and user.

networks, the degrees can be similarly defined. In case a network is undirected, it can be converted to the directed case by simply decomposing every edge to two directed edges with equal weights and opposite directions.

Given the typed essence, an HIN can be abstracted using a network schema $\tilde{G} = (\mathcal{T}, \mathcal{R})$ [14], which provides meta-information regarding the node types and edge types in the HINs. Figure 5.2a gives an example of the schema of a movie reviewing network as an HIN.

Definition 5.2 (Aspect of HIN)**.** *For a given HIN G with network schema $\tilde{G} = (\mathcal{T}, \mathcal{R})$, an* ***aspect*** *of G is defined as a subgraph of the network schema $\tilde{G}$. For an aspect a, we use $\mathcal{T}^a \subseteq \mathcal{T}$ to denote the node types involved in this aspect, and $\mathcal{R}^a \subseteq \mathcal{R}$ as the edge types involved in this aspect .*

As an example, we illustrate two aspects from the schema in Figure 5.2a: one on users' preferences for movies based on genre information (upper part in Figure 5.2b); and the other on the semantics of movies based on the composite information of directors, actors and their countries (lower part in Figure 5.2b). If we denote $\mathcal{A}$ a set of representative aspects generated by a certain method, where information is compatible within each aspect and is not redundant across different aspects, then an HIN with only one aspect will have $|\mathcal{A}| = 1$, $\mathcal{T}^a = \mathcal{T}$, and $\mathcal{R}^a = \mathcal{R}$.

Definition 5.3 (HIN Embedding from Aspects)**.** *Suppose that an HIN $G = (\mathcal{V}, \mathcal{E})$ and a set of representative aspects $\mathcal{A}$ are given. For one aspect $a \in \mathcal{A}$, embedding learning in HIN from one aspect a is to learn a node embedding mapping $f^a : \{u \in \mathcal{V} : \varphi(u) \in \mathcal{T}^a\} \to \mathbb{R}^{d(a)}$, where $d(a)$ is the embedding dimension for a and $d(a) \ll |\mathcal{V}|$. For all aspects in $\mathcal{A}$ and all nodes $u \in \mathcal{V}$, the problem of* ***embedding learning from aspects in HIN*** *is to learn*

corresponding feature vector $\mathbf{f}_u$*, such that* $\mathbf{f}_u = \bigoplus_{a \in \mathcal{A}:\, \varphi(u) \in \mathcal{T}^a} \mathbf{f}_u^a$*, where* $\mathbf{f}_u^a$ *is the embedding of node* u *in aspect* a*.*

We remark that, for nodes of different types, the corresponding $\mathbf{f}_u$ might be of different dimensions by definition.

5.3 THE ASPEM FRAMEWORK

To address the problem of embedding learning from aspects in HIN, we propose a flexible framework to distinguish the semantic information regarding each aspect. Specifically, for a node u, the corresponding embedding vectors $\mathbf{f}_u^a$ are inferred independently for different aspects in $\{a \in \mathcal{A} : \varphi(u) \in \mathcal{T}^a\}$. We name the new framework as AspEm, which is short for Aspect Embedding. AspEm includes three components: (i) selecting a set of representative aspects for the HIN of interest, (ii) learning embedding vectors for each aspect, and (iii) integrating embeddings from multiple aspects. We introduce these components as follows.

5.3.1 Aspect Selection in HINs

Since different aspects are expected to reflect distinct semantic facets of an HIN, an aspect of representative capability should consist of compatible edge types in terms of the information carried by the edges. Therefore, even without supervision from downstream applications, the incompatibility within each aspect can be leveraged to determine the quality of the aspect, and such incompatibility can be inferred from dataset-wide statistics.

Before introducing the proposed incompatibility measure, $\mathrm{Inc}(\cdot)$, we first describe the properties that we posit a proper measure should have as follows.

Property 5.1 (Non-negativity)**.** *For any aspect* a*,* $\mathrm{Inc}(a) \geq 0$*.*

Property 5.2 (Monotonicity)**.** *For two aspects* a_1 *and* a_2*, if* $a_1 \subseteq a_2$*, then* $\mathrm{Inc}(a_1) \leq \mathrm{Inc}(a_2)$*.*

Property 5.3 (Convexity)**.** *For two aspects* a_1 *and* a_2*, if their graph intersection has empty edge set, i.e.,* $\mathcal{E}(a_1 \cap a_2) = \varnothing$*, then* $\mathrm{Inc}(a_1) + \mathrm{Inc}(a_2) \leq \mathrm{Inc}(a_1 \cup a_2)$*.*

We note that the intuition of Property 5.3 is that the incompatibility arises from the co-existence of multiple types of edges. As a result, generating an aspect by the union of a_1 and a_2 could only introduce more incompatibility.

To propose our incompatibility measure, we start from the simplest incompatibility-prone scenario: since the incompatibility arises from the co-existence of edge types, the simplest

incompatible-prone aspects are those with two edge types joined by a common node type. In particular, an aspect in this form can be uniquely determined by a schema-level representation $\varphi_l \xrightarrow{\psi_l} \varphi_c \xrightarrow{\psi_r} \varphi_r$, where $\varphi_l, \varphi_c, \varphi_r \in \mathcal{T}$ are (not necessarily distinct) node types and $\psi_l, \psi_r \in \mathcal{R}$ are edge types. Once the incompatibility measure $\mathrm{Inc}(\cdot)$ is defined for this scenario, it can then be generalized to any aspect a by

$$\mathrm{Inc}(a) \coloneqq \sum_{\langle \varphi_l, \psi_l, \varphi_c, \psi_r, \varphi_r \rangle \subseteq a} \mathrm{Inc}(\varphi_l \xrightarrow{\psi_l} \varphi_c \xrightarrow{\psi_r} \varphi_r), \tag{5.1}$$

where $\langle \varphi_l, \psi_l, \varphi_c, \psi_r, \varphi_r \rangle \subseteq a$ represents enumerating all such sub-aspects in aspect a. For undirected networks, we do not distinguish $\langle \varphi_l, \psi_l, \varphi_c, \psi_r, \varphi_r \rangle$ and $\langle \varphi_r, \psi_r, \varphi_c, \psi_l, \varphi_l \rangle$ in this enumeration process. Note that such generalization meets the criteria in Property 5.2 and 5.3.

Incompatible edge types result in inconsistent information. To reflect such intuition, we define the incompatibility measure on aspects of the form $\varphi_l \xrightarrow{\psi_l} \varphi_c \xrightarrow{\psi_r} \varphi_r$ with a Jaccard coefficient–based formulation over each node of type φ_c – the node type that joins two edge types. Specifically, for node u of type φ_c, we calculate the inconsistency in information observed from ψ_l and ψ_r by

$$\gamma(u) \coloneqq \frac{\sum_{\varphi(\tilde{u})=\varphi_c} \max\left\{\mathbf{P}^{\psi_r}_{u,:}(\mathbf{P}^{\psi_r}_{\tilde{u},:})^\top, \mathbf{P}^{\psi_l^{-1}}_{u,:}(\mathbf{P}^{\psi_l^{-1}}_{\tilde{u},:})^\top\right\}}{\sum_{\varphi(\tilde{u})=\varphi_c} \min\left\{\mathbf{P}^{\psi_r}_{u,:}(\mathbf{P}^{\psi_r}_{\tilde{u},:})^\top, \mathbf{P}^{\psi_l^{-1}}_{u,:}(\mathbf{P}^{\psi_l^{-1}}_{\tilde{u},:})^\top\right\}} - 1, \tag{5.2}$$

where $\mathbf{M}^{\psi_i}$ is the adjacency matrix of edge type ψ_i and $\mathbf{P}^{\psi_i}$ is $\mathbf{M}^{\psi_i}$ after row-wise normalization. We remark that this formulation, with a difference of minus 1, is essentially the inverse of Jaccard coefficient over the one-hop neighbors that u can reach via edge type ψ_l and edge type ψ_r. The inverse is taken since greater Jaccard coefficient implies more similarity while we expect more inconsistency, and the minus 1 is appended so that $\gamma(u) = 0$ when $\mathbf{P}^{\psi_r} = \mathbf{P}^{\psi_l^{-1}}$, *i.e.*, no inconsistency if two edge types are identical. Lastly, we average over all such nodes to find incompatibility score of a simplest incompatible-prone aspect

$$\mathrm{Inc}(\varphi_l \xrightarrow{\psi_l} \varphi_c \xrightarrow{\psi_r} \varphi_r) \coloneqq \frac{1}{|\varphi_c^*|} \sum_{u \in \varphi_c^*} \gamma(u),$$

where φ_c^* is the set of all u in φ_c such that the denominator in Eq. (5.2) is nonzero and $\gamma(u)$ is thereby well-defined. Note that this definition satisfies Property 5.1.

To select a set $\mathcal{A}$ of representative aspects for given HIN under any threshold $\theta \in \mathbb{R}_{\geq 0}$, (i) an aspect with incompatible score greater than θ is not eligible to be selected into $\mathcal{A}$,

because it is not semantically consistent enough; (ii) in case both aspects a_1 and a_2 have incompatible score below θ and $a_1 \subset a_2$, we do not select a_1 into $\mathcal{A}$. We note that the second requirement is intended to keep $\mathcal{A}$ concise, so that the information across different aspects is not redundant. Note that when both computation resource and overfitting in downstream application are not of concern, one may explore the potential of trading in model size for gaining additional performance boost by including both a_1 and a_2 to $\mathcal{A}$.

We will demonstrate by experiments in Section 5.4 that this proposed aspect selection method is effective in the sense that (i) AspEm built atop this method can outperform baselines that do not model aspects; and (ii) the set of aspects selected using this statistics-based unsupervised method can outperform other comparable sets of aspects.

5.3.2 Embedding Learning from One Aspect

To design the embedding algorithm for one aspect, we extend the skip-gram model [44] in an approach inspired by existing network embedding studies [33, 34, 20]. We note that AspEm is a flexible framework that can be directly integrated with other homogeneous network embedding methods [18, 19, 40, 21], other than the adopted skip-gram–based approach, while still enjoying the benefits of modeling aspects in HINs.

For an aspect $a \in \mathcal{A}$, the associated node embeddings can be denoted as $\{\mathbf{f}_u^a\}_{\varphi(u)\in\mathcal{T}^a}$. Recall that $\mathcal{T}^a$ corresponds to the set of node types included in the aspect a. We model the probability of observing edge e with edge type $r \in \mathcal{R}^a$ from node u to node v as

$$p^a(v|u,r) = \frac{\exp\left(\mathbf{f}_u^a \cdot \mathbf{f}_v^a\right)}{\sum_{v'\in\mathcal{V}:\,\varphi(v')=\varphi(v)} \exp\left(\mathbf{f}_u^a \cdot \mathbf{f}_{v'}^a\right)}. \tag{5.3}$$

This equation can be interpreted as the probability of observing v given u and the edge type r. On the other hand, the empirical conditional probability observed from aspect a is

$$\hat{p}^a(v|u,r) = W_{uv}^{(r)} \big/ D_u^{O(r)}. \tag{5.4}$$

To obtain embeddings that reflect the network topology, we seek to minimize the difference between the probability distribution derived from the learned embedding Eq. (5.3) and the empirical probability distribution observed in data Eq. (5.4). Therefore, the embedding learning is reduced to minimizing the following objective function

$$\mathcal{O}^a = \sum_{r\in\mathcal{R}^a} \sum_{u\in\mathcal{V}_{O(r)}} \lambda_u^{(r)} d\big(\hat{p}^a(\cdot|u,r), p^a(\cdot|u,r)\big), \tag{5.5}$$

where $\mathcal{V}_{O(r)} \subseteq \mathcal{V}$ is the set of all nodes with outgoing type-r edges, $\lambda_u^{(r)}$ is the relative importance of node u in the context of edges with type r, and $d(\cdot,\cdot)$ is the KL-divergence. Furthermore, we set $\lambda_u^{(r)} \propto D_u^{O(r)}$ with $\lambda_u^{(r)}$ sum up to 1 for a given edge type r. Putting pieces together, Eq. (5.5) can be rewritten as

$$\mathcal{O}^a = -\sum_{r\in\mathcal{R}^a} \frac{1}{\Omega^{(r)}} \sum_{u\in\mathcal{V}_{O(r)}} W_{uv}^{(r)} \log p^a(v|u,r), \tag{5.6}$$

where $\Omega^{(r)} = \sum_{u,v} W_{uv}^{(r)}$. Consequently, the problem of learning embedding from an aspect $a \in \mathcal{A}$ is equivalent to solving the following optimization problem

$$\min_{\{\mathbf{f}_u^a\}_{u:\varphi(u)\in\mathcal{T}^a}} \mathcal{O}^a. \tag{5.7}$$

With this formulation, information from each aspect of an HIN is transcribed into a different embedding space.

5.3.3 Compositing Node Embedding and Edge Embedding

By solving the optimization problem Eq. (5.7), we are able to obtain a feature vector $\mathbf{f}_u^a$ for each node $u \in \mathcal{V}^a$ from the aspect $a \in \mathcal{A}$, and the final embedding for node u is thereby given by the concatenation of the learned embedding vectors from all aspects involving u, *i.e.*, $\mathbf{f}_u \coloneqq \bigoplus_{a\in\mathcal{A}:\,\varphi(u)\in\mathcal{T}^a} \mathbf{f}_u^a$. To characterize edges for applications such as link prediction, we follow the method in existing work [18] and define the edge embedding mapping g with domain in $\mathcal{V}\times\mathcal{V}$ as $g(u,v) = \mathbf{g}_{uv} \coloneqq \bigoplus_{a\in\mathcal{A}:\,\varphi(u),\varphi(v)\in\mathcal{T}^a} \mathbf{f}_u^a \circ \mathbf{f}_v^a$, where $\circ$ is Hadamard product between two vectors of commensurate dimensions.

Instead of simply focusing on the node embeddings, another important component of networks is the interactions among nodes, *i.e.*, edges. Characterizing edges is important for downstream applications such as link prediction, which aims to predict whether there is an edge between a pair of nodes for a certain edge type. Therefore, it is of interest to define the embedding for edges. In our problem setting, we simply refer to a function of embeddings of a node pair as edge embedding, even if there might be no edge between the given node pair. The function of the edge embeddings is a hyperparameter and can be chosen by various designs.

Multiple possible ways exist in building edge embedding from the embedding vectors of the two involved nodes. In the AspEm framework, we bridge node embedding and edge embedding by Hadamard product [114]. We adopt Hadamard product in this design for two

reasons: (i) For a pair of nodes, the inner product of the node embeddings is equivalent to the sum of Hadmard product of the two embeddings. As formulated in Eq. (4.3), the inner product of the node embeddings plays a vital role in modeling the proximity of edges between nodes. (ii) Empirical experiments on three datasets from a previous study [18] show that Hadamard product is a choice superior to other options in constructing edge embeddings from node embeddings. Specifically, we define the edge embedding mapping g with domain in $\mathcal{V} \times \mathcal{V}$ as $g(u, v) = \mathbf{g}_{uv} \coloneqq \bigoplus_{a \in \mathcal{A}:\, \varphi(u), \varphi(v) \in \mathcal{T}^a} \mathbf{f}_u^a \circ \mathbf{f}_v^a$, where $\circ$ is Hadamard product between two vectors of commensurate dimensions.

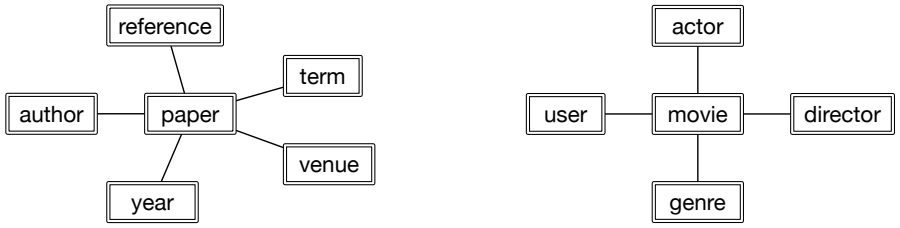
We additionally remark that a recent paper [49] specifically addresses the problem of learning edge representation, and defines edge embedding as a parametric function over node embeddings, which is learned from the dataset. Since the focus of this chapter is not to tackle the problem of edge embedding, we simply adopt the aforementioned straightforward Hadamard approach.

5.3.4 Model Inference

It is computationally expensive to directly optimize the objective function Eq. (5.6) since the partition function in Eq. (5.3) sums over all the nodes in $\mathcal{V}$. Therefore, we approximate it with negative sampling [44] and resort to asynchronous stochastic gradient descent (ASGD) [104] for optimization as with the common practice in skip-gram–based embedding methods [18, 19, 34, 20]. For each iteration in ASGD, we first sample an edge type r from $\mathcal{R}^a$; then sample an edge $e = (u, v)$ of type r with the sampling probability proportional to $W_{uv}^{(r)}$; and finally obtain negative samples from the noise distribution $P_n^{(r)}(v) \propto \left[D_v^{I(r)}\right]^{3/4}$ [44]. The optimization objective for each iteration is therefore $\log \sigma(\mathbf{f}_u^a \cdot \mathbf{f}_v^a) + \sum_{i=1}^{K} \mathbb{E}_{v_i' \sim P_n^{(r)}} \log \sigma(-\mathbf{f}_u^a \cdot \mathbf{f}_{v_i'}^a)$, where $\sigma(\cdot)$ is the sigmoid function $\sigma(x) = \exp(x)/\big(1 + \exp(x)\big)$. This optimization procedure shares the same spirit with some existing network embedding algorithms, and one may refer to the network embedding paper by Tang *et al.* [20] for further details.

5.4 EXPERIMENTS

In order to provide evidence for the efficacy of AspEm, we experiment with two real-world HINs in this section. Specifically, the learned embeddings are fed into two types of downstream applications – multi-class classification and link prediction – to answer the following two questions:



(a) DBLP **(b) IMDb**

Figure 5.3: The network schemas of DBLP and IMDb.

Q1 Does exploiting aspects in HIN embedding learning help better capture the semantics of typed networks in both link prediction and classification tasks?

Q2 Without supervision, is it feasible to select a set of representative aspects just using dataset-wide statistics.

5.4.1 Data Description

We use two publicly available real-world HIN datasets: DBLP and IMDb. **DBLP** is a bibliographical information network in the computer science domain[1]. There are six types of nodes in the network: author (A), paper (P), reference (R), term (T), venue (V), and year (Y), where reference corresponds to papers being referred by other papers. The terms are extracted and released by Chen et al. [29]. The edge types include: author writing paper, paper citing reference, paper containing term, paper publishing in venue, and paper publishing in year. The corresponding network schema is depicted in Figure 5.3a. Note that we distinguish the node type of reference, so that a paper have a different embedding when acting as a reference. **IMDb** is an HIN built by linking the movie-attribute information from IMDb and the user-reviewing-movie relationship from MovieLens-100K.[2] There are five types of nodes in the network: user (U), movie (M), actor (A), director (D), and genre (G). The edge types include: user reviewing movie, actor featuring in movie, director directing movie, and movie being of genre. The network schema can be found in Figure 5.3b. We summarize the statistics of the datasets in Table 5.1.

We use the node types to represent an aspect in these two HINs. For example, APY in the DBLP network refers to the aspect involving author, paper, and year, and UMA in IMDb represents the aspect involving user, movie, and actor. The schema of each aspect can be easily inferred based on the holistic network schema, as shown in Figure 5.3.

[1]https://aminer.org/citation

[2]https://grouplens.org/datasets/movielens/100k/

Table 5.1: Basic statistics for the DBLP and IMDb networks.

DBLP	Author	Paper	Reference	Term	Venue	Year
	1,003,836	1,756,680	693,406	402,687	7,528	62
IMDB	User	Movie	Actor	Director	Genre	
	943	1,360	42,275	918	23	

5.4.2 Baseline Methods and Experiment Setting

To answer Q1 at the beginning of the section, we compare AspEm against several unsupervised embedding methods. **SVD** [115]: a matrix factorization based method, where singular value decomposition is performed on the adjacent matrix of the homogeneous network and the first d singular vectors are taken as the node embeddings of the network, where d is the dimension of the embedding. **DeepWalk** [19]: a homogeneous network embedding method, which samples multiple walks starting from each node, and then applies the skip-gram model to learn embedding. **LINE** [20]: a homogeneous network embedding method, which treats the neighbors of a node as its context, and then applies the skip-gram model to learn embedding. **OneSpace**: as a heterogeneous network embedding method, OneSpace serves as a direct comparison against the proposed AspEm algorithm to validate the utility of embedding different aspects into multiple spaces. This method is given by the proposed AspEm framework with the full HIN schema as the only selected aspect. We note that the OneSpace method embeds all nodes into only one low-dimensional space. In the special case of HINs with star-schema, OneSpace is identical to PTE proposed in [34]. We remark that DeepWalk is identical to node2vec [18] under default hyperparameters.

For the baselines developed for homogeneous networks, we treat the HIN as a homogeneous network by neglecting the node types. Additionally, we apply the same downstream learners onto the embeddings yielded by different embedding methods for fair comparison.

Parameters. While AspEm is capable of using different dimensions for different aspects, we employ the same dimension for all aspects out of simplicity. In other words, we set $d(a_1) = \ldots = d(a_{|\mathcal{A}|}) = d, a_1, \ldots, a_{|\mathcal{A}|} \in \mathcal{A}$. In particular, we set $d = 100$ for DBLP and $d = 10$ for IMDb. For fair comparison, we experiment with two dimensions for every baseline method: (i) the dimension of one aspect used by AspEm (*i.e.*, d) and (ii) the total dimension of all aspects employed by AspEm (*i.e.*, $|\mathcal{A}| \cdot d$). We report the better result between the two choices of dimension for every baseline method. $1,000$ million edges are sampled to learn the embedding on DBLP, and 100 million edges are sampled on IMDb. The number of negative samples is set to 5 following the common practice in network embedding [20].

Selected aspects. Since all our experiments on DBLP involve the node type author (A),

Table 5.2: Classification accuracy in two DBLP tasks.

Dataset/task	DBLP-group		DBLP-area	
Classifier	LR	SVM	LR	SVM
SVD	0.7566	0.7550	0.8158	0.8008
DeepWalk	0.6629	0.7077	0.8308	0.8390
LINE	0.7037	0.7314	0.8526	0.8540
OneSpace	0.7685	0.8333	0.8758	0.8731
AspEm	**0.8425**	**0.8889**	**0.8786**	**0.8813**

we set the threshold for incompatibility measure θ to be the *smallest possible value* such that all node types co-exist with the node type author (A) in at least one aspect eligible to be selected to $\mathcal{A}$ as per the two requirements discussed in Section 5.3.1. As a result, θ is set to be 221267 on DBLP, and the set of selected representative aspects, $\mathcal{A}$, is {APRTV, APT}. Similarly for IMDb, considering that all its experiments involve the node type user (U), θ is set to be 1927.68, and the set of selected representative aspects, $\mathcal{A}$, is {UMA, UMD, UMG}.

The detailed presentation on the calculations and figures involving threshold and aspect selection for both HINs are provided later in this section.

5.4.3 Classification

For classification tasks, we use the learned embeddings as node features and then classify the nodes into different categories using off-the-shelf classifiers. The classification performance is evaluated using accuracy. For a set of concerned nodes $\mathcal{X}$ and node $x \in \mathcal{X}$, denote $l(x)$ the predicted label of x and denote $l^*(x)$ the ground truth label. Then accuracy is defined as Acc. $= \frac{1}{|\mathcal{X}|}\sum_{x \in cX} \delta\big(l(x) = l^*(x)\big)$, where $|\mathcal{X}|$ is the cardinality of $\mathcal{X}$ and $\delta(\cdot)$ is the indicator function.

Due to the availability of trustworthy class labels, we perform two classification tasks on DBLP. The first one (**DBLP-group**) is on the research group affiliation of each author. We consider four research groups led by Christos Faloutsos, Dan Roth, Jiawei Han, and Michael I. Jordan. 116 authors in the dataset are labeled with such group affiliation. The second label set (**DBLP-area**) is on the primary research area of authors. 4,040 authors are manually labeled in four research areas: data mining, database, machine learning, and artificial intelligence [15].

We experiment with two widely used classifiers. One is logistic regression (LR) and the other is support vector machine (SVM). Both classifiers are based on the liblinear imple-

Table 5.3: Link prediction results on DBLP and IMDb.

Dataset	DBLP						IMDb					
Metrics	$P@1$	$P@3$	$P@10$	$R@1$	$R@3$	$R@10$	$P@1$	$P@3$	$P@10$	$R@1$	$R@3$	$R@10$
SVD	0.6648	0.5164	0.2274	0.2939	0.6178	0.8512	0.2470	0.2474	0.2249	0.0152	0.0445	0.1343
DeepWalk	0.7395	0.5297	0.2303	0.3268	0.6329	0.8622	0.3499	0.3605	0.3416	0.0253	0.0774	0.2236
LINE	0.7404	0.5367	0.2299	0.3267	0.6375	0.8596	0.4782	0.4701	0.4130	0.0379	0.1133	0.3137
OneSpace	0.7440	0.5381	0.2279	0.3301	0.6401	0.8519	0.4665	0.4386	0.3852	0.0435	0.1146	0.3038
AspEm	**0.7724**	**0.5645**	**0.2356**	**0.3479**	**0.6749**	**0.8810**	**0.5090**	**0.4853**	**0.4219**	**0.0464**	**0.1296**	**0.3420**

mentation.[3] The classification accuracy for different methods are reported in Table 5.2.

The proposed AspEm method outperformed all four baselines in both tasks with either of the two downstream learners applied. In particular, AspEm yielded better results than OneSpace, which confirms our intuition that there exists incompatibility among aspects, and learning node embeddings independently from different aspects can better preserves the semantics of an HIN. In addition, we observed that the classification results of AspEm were significant better than OneSpace in research group classification; while the improvement of AspEm over OneSpace was less significant in research area classification. This can be partially explained by that the label of research groups is more relevant to temporal information compared with that of research area, and the presence of the aspect APY in AspEm may therefore be more informative for the research group classification task.

Based on the results in Table 5.2, another observation is that the embedding methods distinguishing node types (OneSpace and AspEm) performed better than those not considering node types. This observation is in line with previous studies [33], and can be explained by the heterogeneity of node types in HINs. The nodes of different types in HINs have different properties, such as degrees distribution. Simply ignoring such information can lead to information loss.

5.4.4 Link Prediction

On experiments with link prediction essence, we perform author identification on the DBLP dataset, and user review prediction on the IMDb dataset. Precision and recall are used for evaluating these tasks. Precision at k ($P@k$) is defined as $P@k = \frac{\text{\# of true instances at top } k}{k}$, and recall at k ($R@k$) is defined as $R@k = \frac{\text{\# of true instances at top } k}{\text{\# of total true instances}}$.

We describe the key facts on deriving features for link prediction as follows. **DBLP** – The author identification task on DBLP aims at re-identifying the authors of an anonymized paper, where the reference, term, venue, and year information is still available. Since papers in the test set do not appear in the training set, their embeddings are hence not available.

[3]https://www.csie.ntu.edu.tw/ cjlin/liblinear/

Table 5.4: Link prediction results (P@1) using only one edge.

Edge embedding used	AR	AT	AV	AY
Aspect APRTVY (OneSpace)	0.6933	0.6723	0.6501	0.3166
Aspect APRTV	**0.7566**	**0.6977**	**0.6878**	– –
Aspect APR	0.6071	——	——	——
Aspect APT	——	0.6802	——	——
Aspect APV	——	——	0.5836	——
Aspect APY	——	——	——	0.3187

Therefore, we use the edge embedding of an author and each attribute of a paper (reference, term, venue, or year) to infer whether this author writes this paper. Specifically, for both train and test sets, we derive the feature of an author–paper pair by (i) first computing the edge embedding of the concerned author and each attribute of the concerned paper; (ii) then averaging all edge embedding vectors with the same edge type (author–reference, author–term, author–venue, or author–year) to find four edge-type-specific vectors; (iii) finally deriving the feature vector for an author–paper pair by concatenating of the previous four averaged edge embedding vectors. We randomly selected 32,488 papers into the test set, and take the rest as training data. Following the procedure proposed by Chen et al. [29], for each paper in test, we randomly sample a set of negative authors, which together with all the true authors constitute the candidate author set of size 100. **IMDb** – The user review prediction task on IMDb aims at predicting which user reviews a movie. Features for user–movie pairs are likewise derived as with author–paper pairs in DBLP. As with the DBLP author identification task, we sampled a candidate set of 100 movies for each user for testing on DBLP.

On top of the derived node pair features as well as labels in the training set, logistic regression is trained for inferring the existence of edges in the test set. We choose the scikit-learn[4] implementation with the SAG solver for logistic regression – different from that used for classification – because the SAG solver converges faster than liblinear, and the author identification task on DBLP has a huge number of author–paper pairs as training instances.

From the main results on link prediction presented in Table 5.3, we have observation consistent with the classification tasks that OneSpace and AspEm had better performance than the methods without considering type information. Also, AspEm outperformed OneSpace.

Predictive power of single edge embedding. In order to better understand the mechanism of AspEm in the link prediction tasks, we dissect each aspect and study the predictive power of a single edge embedding from one aspect. Specifically, we use each edge embedding

[4]http://scikit-learn.org/stable/

over an author-attribute pair from one aspect for link prediction. Due to space limitation, we focus on the link prediction task on DBLP, because it has the largest number of available labels and can thereby yield most reliable conclusions. The experimental results are presented in Table 5.4, where the rows correspond to the aspect being used for embedding learning and the columns correspond to the edge embedding being used for link prediction.

It can be seen that using the aspect APRTV was better than using the bigger aspect APRTVY for all edge embeddings, where APRTVY was identical to the whole network schema. Such result provides evidence that for certain HIN datasets, using all the information in the network may be less effective than using partial information (i.e., one aspect). We interpret this result as: on the one hand, an author may focus on certain research field that cites certain classic references (R), uses certain terminologies (T), and publishes papers in certain venues (V), *i.e.*, R, T, and V correlate to some extent; on the other hand, an author may be actively publishing papers in a certain range of years (Y). However, the information regarding R, T, and V do not align well with Y. As a result, embedding R, T, V, and Y together into the same space (as in the OneSpace model) led to worse embedding quality even though more types of data were used. This result further consolidated our insight that HIN can have multiple aspects, and one should embed aspects with different semantics into distinct spaces.

To conclude, the results for classification and link prediction give an affirmative answer to Q1 – Distinguishing the information from semantically different aspects can benefit HIN embedding learning.

5.4.5 The Impact of Aspect Selection

In the previous section, we have shown that the aspect selection method proposed in Section 5.3.1 can effectively support the AspEm framework to outperform embedding methods that do not model aspects in HINs. In this section, we further address Q2 and demonstrate the set of representative aspects AspEm selected using the proposed method is of good quality compared with other selections of aspects.

To this end, we again use the link prediction on DBLP as the downstream evaluation task, and experiment with all sets of aspect that are comparable to {APRTV, APY}. Specifically, each of these comparable sets of aspects (i) has two aspects, and (ii) author and paper appear in both aspects, and other node types exist in exactly one of the two aspects.

From the results presented in Table 5.5, it can be seen that the set of representative aspects selected by our proposed method, {APRTV, APY}, achieved the best performance among all comparable aspect selections. Note that all the 6 inferior sets of aspects have

Table 5.5: Link prediction results using different 2-combinations aspects on DBLP.

Metrics	P@1	P@3	P@10	R@1	R@3	R@10
{APTV, APRY}	0.7522	0.5476	0.2303	0.3362	0.6524	0.8611
{APRV, APTY}	0.7347	0.5327	0.2257	0.3271	0.6327	0.8425
{APRT, APVY}	0.7579	0.5556	0.2332	0.3385	0.6614	0.8708
{APTVY, APR}	0.7384	0.5360	0.2277	0.3280	0.6372	0.8499
{APRVY, APT}	0.7353	0.5356	0.2271	0.3263	0.6355	0.8474
{APRTY, APV}	0.7366	0.5362	0.2277	0.3274	0.6364	0.8492
{APRTV, APY}	**0.7724**	**0.5645**	**0.2356**	**0.3479**	**0.6749**	**0.8810**

inconsistency score, Inc($\cdot$), greater than the threshold we set, which can be verified from the numbers provided in Section 5.4.7. This result further consolidates the feasibility of selecting representative aspects for any HIN solely by dataset-wide statistics without the need of additional task-specific supervision.

5.4.6 Hyperparameter Study

We vary two hyperparameters, one at each time, that play important roles in embedding learning: dimension of embedding spaces and the number of edges sampled in the training phase. All other parameters are set following Section 5.4.2.

The performance in the link prediction task on DBLP is presented in Figure 5.4. It can be seen that model performance tended to be better as either the dimension of embedding spaces or the number of edges sampled grew, and the growth became less drastic after dimension reached 100 and number of edges sampled reached 1000 million. Such a pattern agrees with the results in other similar studies [18, 33, 20].

5.4.7 Incompatibility Score of Each Aspect in DBLP and IMDb

In this section, we provide the sufficient statistics for calculating incompatibility of each aspect as defined in Section 4.1 from the main content of the paper. That is, we provide the incompatibility of aspects of the form $\varphi_l \xrightarrow{\psi_l} \varphi_c \xrightarrow{\psi_r} \varphi_r$ as in Table 5.6 for DBLP and Table 5.7 for IMDb. Note that the proposed AspEm framework selects a set of representative aspects $\mathcal{A}$ for embedding purpose based on their incompatibility, which will be illustrated in the next section.

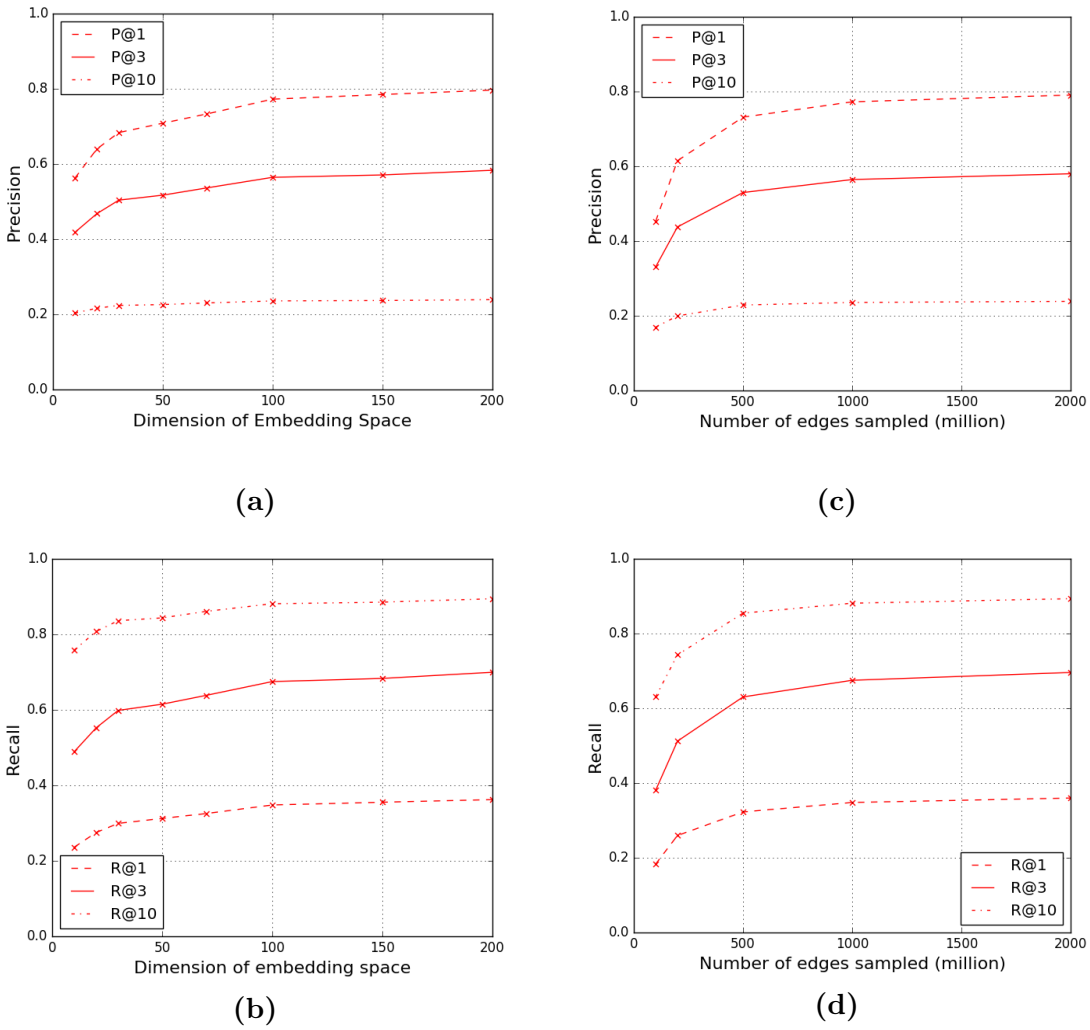


Figure 5.4: (a) and (b) depict the precision and recall against various dimensions employed for the embedding space. (c) and (d) give the precision and recall against various choices of sampled edge numbers.

5.4.8 Aspect Selection in DBLP and IMDb

Using Eq. (4.1) and the sufficient statistics provide in Table 5.6 and 5.7, one can calculate the incompatibility score of any aspect in DBLP and IMDb. We proceed to illustrate the aspect selection results using DBLP as example.

Given any threshold $\theta \in \mathbb{R}_{\geq 0}$, (i) any aspect with incompatible score greater than θ is not eligible to be selected into $\mathcal{A}$, because it is not meaningful and semantically consistent enough to be one representative aspect of the involved HIN; (ii) in case both aspects a_1 and a_2 have incompatible score below θ and $a_1 \subset a_2$, we do not select a_1 into $\mathcal{A}$. We note that the second requirement is intended to keep $\mathcal{A}$ concise and representative in the aspect selection process. However, when both computation resource and overfitting in downstream application are

Table 5.6: Sufficient statistics for incompability on DBLP.

Aspect	Incompatibility score
$R-P-Y$	52753.6
$A-P-Y$	221267.
$T-P-Y$	10254.4
$V-P-Y$	1830.08
$A-P-R$	307.988
$T-P-R$	6060.62
$V-P-R$	948.654
$T-P-A$	11518.2
$V-P-A$	5724.80
$V-P-T$	3579.59

Table 5.7: Sufficient statistics for incompability on IMDb.

Aspect	Incompatibility score
$U-M-A$	171.607
$D-M-A$	1689.76
$G-M-A$	12956.6
$D-M-U$	1927.68
$G-M-U$	636.442
$G-M-D$	531.266

not of concern, one may explore the possibility of gaining additional performance boost by adding both a_1 and a_2 to $\mathcal{A}$.

Aspects in DBLP satisfying the aforementioned two requirements at various threshold θ are presented in Figure 5.5. Since all our experiments on DBLP involve the node type author (A), we set θ to be the *smallest possible value* such that all node types co-exist with the node type author (A) in at least one aspect eligible to be selected to $\mathcal{A}$ as per the aforementioned two requirements. Therefore, θ is set to be 221267 on DBLP. One can verify this by calculating $\text{Inc}(APRTV) = \text{Inc}(APR)+\text{Inc}(TPA)+\text{Inc}(VPA)+\text{Inc}(TPR)+\text{Inc}(VPR)+\text{Inc}(VPT) = 28139.9$, $\text{Inc}(APY) = 221267$, and $\text{Inc}(YPRTV) = \text{Inc}(RPY) + \text{Inc}(TPY) + \text{Inc}(VPY) + \text{Inc}(TPR) + \text{Inc}(VPR) + \text{Inc}(VPT) = 75426.9$.

Furthermore, aspects not involving author (A) are additionally excluded from $\mathcal{A}$ (those outside of the dotted boxes in Figure 5.5), because whether or not adding them to $\mathcal{A}$ does not affect the downstream evaluations. As a result, the set of selected representative aspects, $\mathcal{A}$, for DBLP is {APRTV, APT}.

Similarly for IMDb, following the same requirements and the consideration that all its experiments involve the node type user (U), θ is set to be 1927.68, and the set of selected

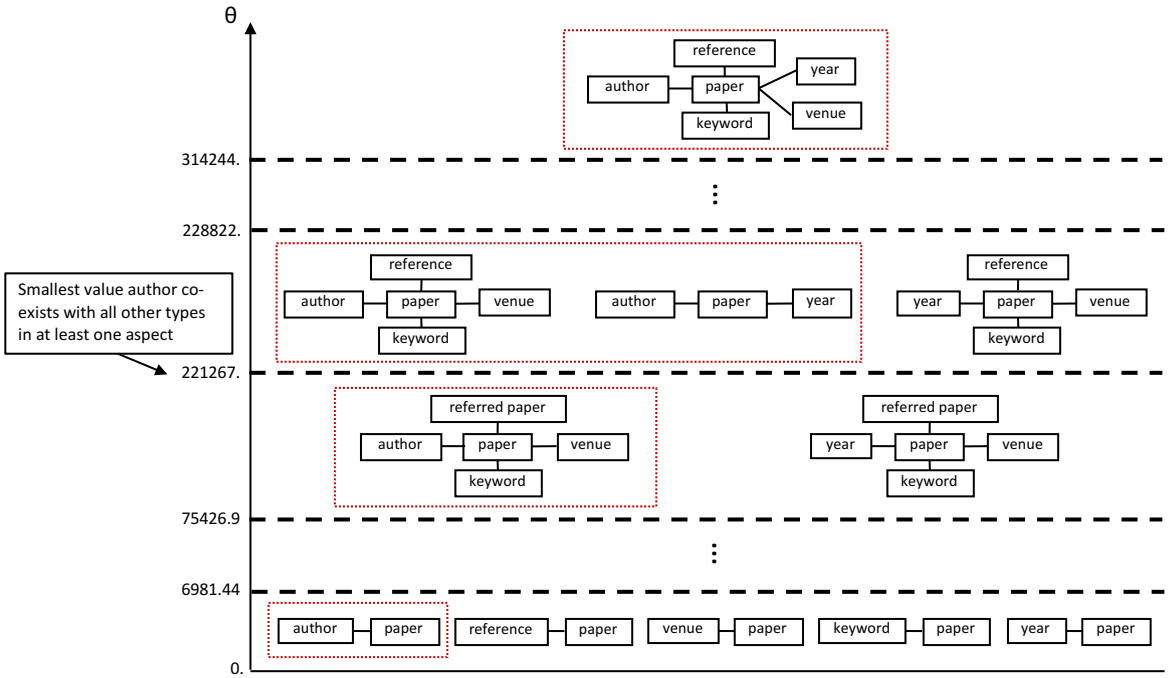


Figure 5.5: Aspects in DBLP satisfying the two requirements at various threshold θ.

representative aspects, $\mathcal{A}$ is {UMA, UMD, UMG}.

5.5 SUMMARY

In our work, we study the problem of embedding learning in HINs. Particularly, we make the key observation that there are multiple aspects in heterogeneous information networks and there might be incompatibility among different aspects. Therefore, we take advantage of the information encapsulated in each aspect and propose AspEm– a new embedding learning framework from aspects, which comes with an unsupervised method to select a set of representative aspects from an HIN. We conducted experiments to corroborate the efficacy of AspEm in better representing the semantic information in HINs.

CHAPTER 6: HARNESSING HETEROGENEOUS ASSOCIATION WITH HETEROGENEOUS METRICS

6.1 OVERVIEW

Heterogeneous information networks (HINs) have received increasing attention in the past decade due to its ubiquity and capability of representing rich information [2, 14]. Meanwhile, network embedding has emerged as a scalable representation learning method [55, 18, 19, 40, 34, 20, 21]. Network embedding learns low-dimensional vector representations for nodes to encode their semantic information in the original network. The vectorized representations can be easily combined with off-the-shelf machine learning algorithms for various tasks such as classification and link prediction [19, 18, 20, 41], which provides a convenient approach for researchers and engineers to mine and learn from the networked data. To marry the advantages of HINs and network embedding, researchers have recently started to explore methods to embed heterogeneous information networks [56, 55, 54, 32, 33, 34, 57], and have demonstrated the effectiveness of HIN embedding in applications including author identification [29], name disambiguation [116], proximity search [58], event detection [117], *etc.*

However, the heterogeneity in HINs brings in not only rich information but also potentially incompatible semantics, which poses special challenges to embed heterogeneous information networks. Take the movie-reviewing network in Figure 6.1 as an example, where users review movies and list certain actors, directors, and genres as their favorites. Suppose user *Stan* likes both movies directed by *Ang Lee* (director) and musical (genre). Since *Ang Lee* has never directed any musical, nor is he semantically similar to musical, if this HIN were embedded into *one* metric space, musical and *Ang Lee* would be distant from each other, while the user *Stan* would not be simultaneously close to both of them, due to the triangle inequality property of metric spaces. We have also observed different extents of such incompatibility from real-world data as to be discussed in Section 6.3, which is consistent with the observation that different extents of correlation can exist within one HIN as per existing study [38]. As a result, it can be expected that an algorithm would generate better embeddings if it additionally models such semantic incompatibility. We hence study the problem of *comprehensive transcription of heterogeneous information networks*, which purely aims to transcribe the rich and potentially incompatible information from HINs to the embeddings, without involving additional expertise, feature engineering, or installation of supervision.

With HINs comprehensives transcribed, one can again pipe the *unsupervisedly* learned

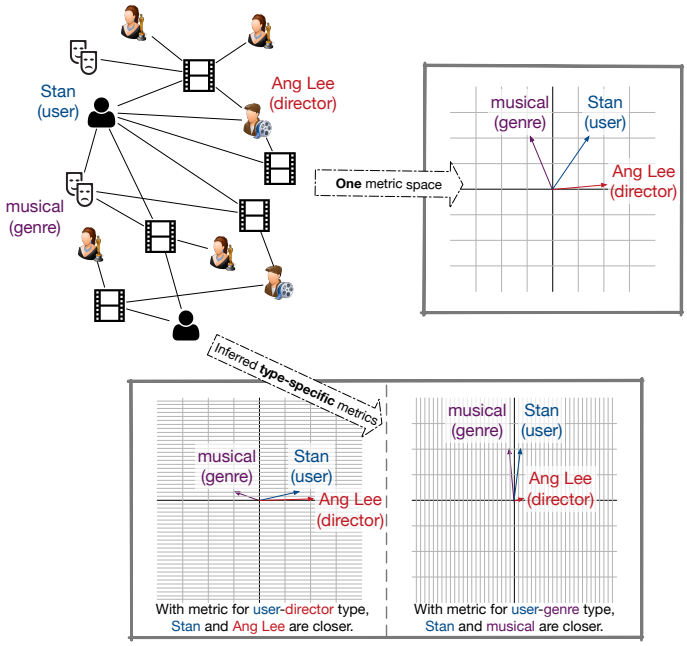


Figure 6.1: To preserve the rich information in HIN embedding, properly handling the incompatibility introduced by the heterogeneity is necessary. The upper left part of the figure gives a toy movie-reviewing HIN, where users review movies and list certain directors, actors, genres as their favorites. *Stan* likes both musical and movies directed by *Ang Lee*. If all nodes were embedded to one metric space, *Stan* would be close to neither musical nor *Ang Lee* due to the dissimilarity between musical and *Ang Lee*. This results in information loss in the embedding learning process. However, we can alleviate this problem by employing edge representation and inferring edge-type–specific metrics, so that *Stan* can be close to both musical and *Ang Lee* under their respective metrics, while not necessarily dragging musical and *Ang Lee* closer. The two metrics shown in the lower figure can be achieved by linearly transforming the metric space in the upper right figure.

embeddings to off-the-shelf machine learning algorithms for a wide range of applications. Therefore, beyond the capability of preserving rich information, another motivation to study comprehensive transcription of HINs is to provide an easy-to-use approach to unleash the power of HINs in a wide variety of applications with no expertise or supervision required in the embedding learning process.

Traditional homogeneous network embedding methods [18, 19, 40, 20, 21] treat all the nodes and edges equally regardless of their types, which do not capture the essential heterogeneity of HINs. A couple of methods have recently been studied for embedding heterogeneous information networks [56, 55, 54, 32, 33, 34, 57]. Many of them build their algorithms on top of a set of meta-paths [56, 55], which often require users to specify the meta-paths or

leverage supervision to make the meta-path selection. However, a set of meta-paths specified or selected in this way often only reflects certain aspects of the HIN or is suitable for specific tasks. As a result, they are not always capable of transcribing HINs comprehensively. These methods are not as easy-to-use either because it involves the additional meta-path generation process that entails expertise or supervision. Besides using meta-paths, some approaches have been proposed to embed specific kinds of HINs [33, 34] for certain tasks or HINs with additional side information [32]. These methods cannot be applied to comprehensively transcribe general HINs. Additionally, most existing HIN embedding methods [56, 55, 33, 34] employ only one metric space for embedding learning. This approach may suit downstream tasks that are related to certain partial information of an HIN with compatible semantics but could lead to information loss if the objective is to comprehensively transcript the entire HIN.

The problem of comprehensive transcription of HINs is challenging because it requires the modeling of heterogeneity that can be complex and incompatible. Besides, without the availability of supervision, proposed solutions need to capture the latent structure of the HINs and distinguish potentially incompatible semantics in an unsupervised way. To cope with these challenges, we propose **h**eterogeneous information network **e**mbedding via **e**dge **r**epresentations, which is henceforth referred to as HEER. HEER builds edge embeddings atop node embeddings, which are further coupled with inferred heterogeneous metrics for each edge type. The inferred metrics capture which dimensions of the edge embeddings are more important for the semantic carried by their corresponding edge types. In turn, the information carried by edges of different types updates the node embeddings and edge embeddings with emphases on different type-specific manifolds. In this way, we can preserve different semantics even in the presence of incompatibility. Still take the movie-reviewing network as example, by adopting heterogeneous metrics as in the lower part of Figure 6.1, *Stan* could be close to both musical(genre) and *Ang Lee*(director) under their respective metrics. Furthermore, the heterogeneous metrics are inferred by fitting the input HIN, so that semantic incompatibility is captured without additional supervision.

Specifically, with the availability of edge representations and coupled metrics, we derive loss function that reflects both the existence and the type of an edge. By minimizing the loss, the node embeddings, edge embeddings, and heterogeneous metrics are updated simultaneously, and thereby retain the heterogeneity in the input HIN. Different extents of incompatibility can also be modeled, where the more compatible two edge types are, the more similar their corresponding metrics would be.

Lastly, we summarize our contributions as follows:

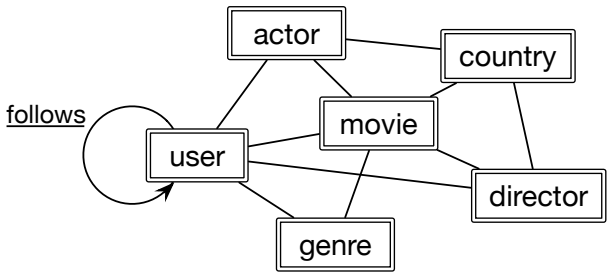


Figure 6.2: The schema of a toy movie-reviewing HIN with six node types, seven undirected edge types, and one directed edge type.

1. We propose to study the problem of comprehensive transcription of HINs in embedding learning, which preserves the rich information in HINs and provides an easy-to-use approach to unleash the power of HINs.

2. We identify that different extents of semantic incompatibility exist in real-world HINs, which pose challenges to the comprehensive transcription of HINs.

3. We propose an algorithm, HEER, for the comprehensive transcription of HINs that leverages edge representations and heterogeneous metrics.

4. Experiments with real-world large-scale datasets demonstrate the effectiveness of HEER and the utility of edge representations and heterogeneous metrics.

6.2 PRELIMINARIES

In this section, we define related concepts and notations.

Definition 6.1 (Heterogeneous Information Network)**.** *An* ***information network*** *is a directed graph $G = (\mathcal{V}, \mathcal{E})$ with a node type mapping $\varphi : \mathcal{V} \to \mathcal{T}$ and an edge type mapping $\psi : \mathcal{E} \to \mathcal{R}$. Particularly, when the number of node types $|\mathcal{T}| > 1$ or the number of edge types $|\mathcal{R}| > 1$, the network is called a* ***heterogeneous information network*** *(HIN).*

Given the typed essence of HINs, the network schema $\tilde{G} = (\mathcal{T}, \mathcal{R})$ [14] is used to abstract the meta-information regarding the node types and edge types in an HIN. Figure 6.2 illustrates the schema of a toy movie-reviewing HIN.

In addition, we require that only one node type can be associated with a certain end of an edge type. That is, once an edge type is given, we would deterministically know the node types on its two ends. As an example, consider two edges with one representing director *Fatih Akin* living in *Germany* and another representing movie *In the Fade* being produced

in *Germany*. Such requirement implies that these two edges must have distinct types – *livesIn* and *isProducedIn* – instead of just one type – *isIn*. For edge type $r \in \mathcal{R}$, we denote $\mathcal{P}^r \coloneqq \big\{(u,v) \in \mathcal{V} \times \mathcal{V} \,\big|\, \varphi(u) \sim r \sim \varphi(v))\big\}$, where $\varphi(u) \sim r \sim \varphi(v)$ means the node type pair $(\varphi(u), \varphi(v)))$ is consistent with edge type r. Additionally, define $\mathcal{P}^r_{u*} \coloneqq \big\{\tilde{v} \in \mathcal{V} \big| (u,\tilde{v}) \in \mathcal{P}^r\big\}$ and $\mathcal{P}^r_{*v} \coloneqq \big\{\tilde{u} \in \mathcal{V} \big| (\tilde{u},v) \in \mathcal{P}^r\big\}$.

Moreover, when the network is weighted and directed, we use $W^{(r)}_{uv}$ to denote the weight of an edge $e \in \mathcal{E}$ with type $r \in \mathcal{R}$ that goes out from node u toward v. $D^{O(r)}_u$ and $D^{I(r)}_u$ respectively represent the outward degree of node u (*i.e.*, the sum of weights of all type-r edges going outward from u) and the inward degree of node u (*i.e.*, the sum of weights of all type-r edges going inward to u). For an unweighted edge, $W^{(r)}_{uv}$ is trivially 1. For an undirected edge, we always have $W^{(r)}_{uv} = W^{(r)}_{vu}$ and $D^{O(r)}_u = D^{I(r)}_u$.

Definition 6.2 (Node and Edge Representations in HIN Embedding)**.** *Given an HIN $G = (\mathcal{V}, \mathcal{E}; \varphi, \psi)$, the problem of HIN embedding via edge representations learns a node embedding mapping $f : \mathcal{V} \to \mathbb{R}^{d_\mathcal{V}}$ and an edge embedding mapping $f : \mathcal{V} \times \mathcal{V} \to \mathbb{R}^{d_\mathcal{E}}$, where $d_\mathcal{V}$ and $d_\mathcal{E}$ are the dimensions for node and edge embeddings, respectively. A node $u \in \mathcal{V}$ is thereby represented by a node embedding $\mathbf{f}_u \coloneqq f(u)$ and a node pair $(u,v) \in \mathcal{V} \times \mathcal{V}$ is represented by an edge embedding $\mathbf{g}_{uv} \coloneqq g(u,v)$.*

With this definition, a node pair has its edge embedding even if no edge of any type has been observed between them. On the other hand, it is possible for node pair (u,v) to be associated by multiple edges with different types, and we expect edge embedding $\mathbf{g}_{uv}$ to encapsulate such information of an HIN.

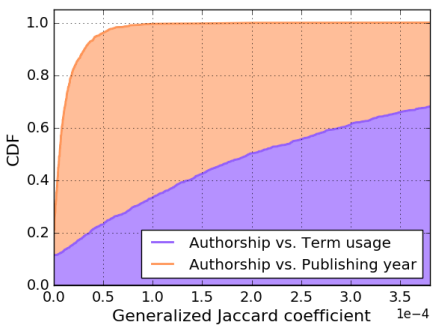
Finally, we define the problem of comprehensive transcription of a heterogeneous information network in embedding learning.

Definition 6.3 (Comprehensive Transcription of an HIN)**.** *The comprehensive transcription of an HIN aims to learn the representations of the input HIN that retains the rich in the HIN as comprehensively as possible, in an approach that does not require additional expertise, feature engineering, or supervision.*

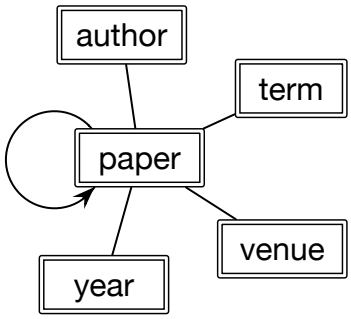
6.3 VARIED EXTENTS OF INCOMPATIBILITY DUE TO HETEROGENEITY

In this section, we look into the incompatibility in HINs using real-world data, and we take DBLP as an example.

DBLP is a bibliographical information network in the computer science domain [118], where authors write papers that are further associated with nodes of other attribute types.



(a) The CDF of the generalized jaccard coefficients for two pairs of edge types.



(b) The schema of the DBLP network.

Figure 6.3: Varied extents of incompatibility exist between different pairs of edge types in the DBLP network.

Since the measurable incompatibility in an HIN arises from the co-existence of multiple edge types, we dive down to the minimal case that involves two different edge types (r_1 and r_2) joined by a common node type (t). To quantify the incompatibility for this minimal case, we use the widely used generalized Jaccard coefficient to measure the similarity between the node groups reachable from a given node of type t via the two edge types. Specifically, given node u of type t, the Jaccard coefficient for edge types r_1 and r_2 is given by $J(u; r_1, r_2) := \frac{\min_{\varphi(v)=t}\{l(u,v;r_1),l(u,v;r_2)\}}{\max_{\varphi(v)=t}\{l(u,v;r_1),l(u,v;r_2)\}}$, where $l(u, v; r) := \boldsymbol{P}^r_{u,:}(\boldsymbol{P}^r_{v,:})^\top$ is the reachability between nodes u and v via edge type r and $\boldsymbol{P}^r$ is the row-normalized adjacency matrix of edge type r. Generalized Jaccard coefficient has a range of $[0, 1]$, and greater value implies more similarity, or equivalently, less incompatibility.

As an example, we consider four node types – author, paper, key term, and year – and two pairs of edge types – (i) authorship vs. publishing year of papers and (ii) authorship vs. term usage of papers. We illustrate the distributions over Jaccard coefficient using cumulative distribution function (CDF) for each of the two pairs in Figure 6.3a. It can be seen that over 95% of nodes have a generalized Jaccard coefficient smaller than $5e^{-5}$ between authorship and publishing year, while less than 25% of nodes fall in the same category when it comes to authorship vs. term usage. In other words, we observe more incompatibility between authorship and publishing year than between authorship and term usage. However, this relationship is actually not surprising because papers published in the same year can be authored by any researchers who are active at that time, while key terms associated to certain research topics are usually used by authors focusing on these topics. With the presence of such varied extent of incompatibility, we would expect an embedding algorithm

tailored for comprehensive transcription of HINs to be able to capture this semantic subtlety in HINs.

In fact, by employing edge representation and heterogeneous metrics, the inferred metrics could be learned to be different for incompatible edge types. In turn, the information carried by these two edge types would be updating the node embeddings and edge embeddings with emphases on different manifolds. On the other hand, the subtlety of the different extent of incompatibility could also be captured in a way that the more compatible two edge types are, the more similar their inferred metrics should be.

6.4 PROPOSED METHOD

To provide an general-purpose, easy-to-use solution to HIN embedding, we describe the HEER model in this section, where HEER stands for **H**eterogeneous Information Network **E**mbedding via **E**dge **R**epresentations. Afterward, the model inference method is described subsequently.

6.4.1 The HEER Model

A learned embedding that effectively encodes the semantics of an HIN should be able to reconstruct this HIN. With the use of edge representation, we expect the embedding to infer not only the existence but also the type of edge between each pair of nodes. For edge type $r \in \mathcal{R}$, we formulate the **typed closeness** of node pair (u, v) atop their edge embedding $\mathbf{g}_{uv}$ as

$$s_r(u,v) \coloneqq \begin{cases} \dfrac{\exp(\boldsymbol{\mu}_r^\top \mathbf{g}_{uv})}{\sum\limits_{\tilde{v} \in \mathcal{P}^r_{u*}} \exp(\boldsymbol{\mu}_r^\top \mathbf{g}_{u\tilde{v}}) + \sum\limits_{\tilde{u} \in \mathcal{P}^r_{*v}} \exp(\boldsymbol{\mu}_r^\top \mathbf{g}_{\tilde{u}v})}, & (u,v) \in \mathcal{P}^r, \\ 0, & (u,v) \notin \mathcal{P}^r, \end{cases} \tag{6.1}$$

where $\boldsymbol{\mu}_r \in \mathbb{R}^{d_\mathcal{V}}$ is a edge-type–specific vector to be inferred that represents the metric coupled with this type. Edge types with compatible semantics are expected to share similar $\boldsymbol{\mu}_r$, while incompatible edge types make use of different $\boldsymbol{\mu}_r$ to avoid the embedding learning on respective semantics to dampen each other.

To measure the capability of the learned embedding in reconstructing the input HIN, the difference between the observed weights and the typed closeness inferred from embedding

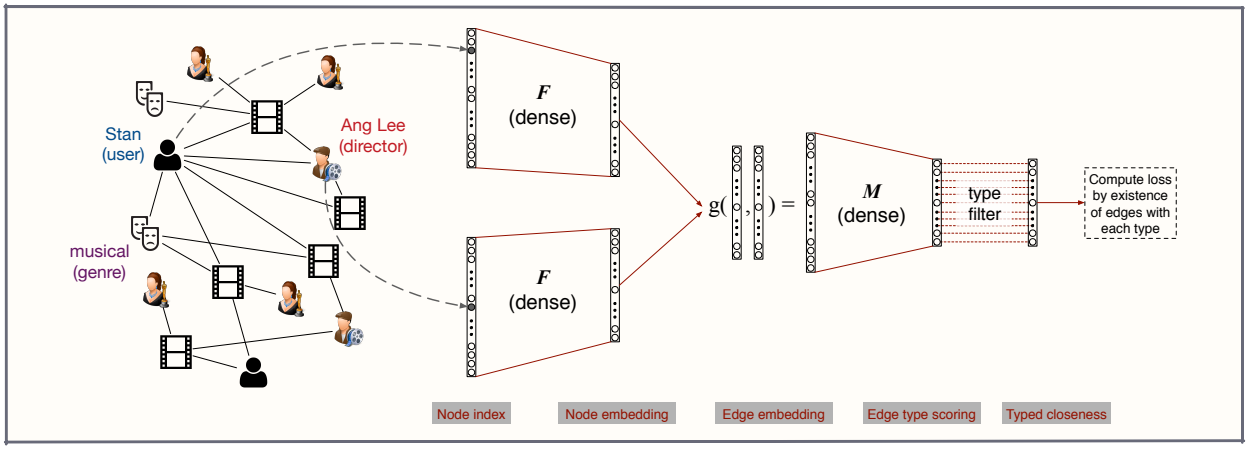


Figure 6.4: An illustration of the HEER architecture for learning HIN embedding via edge representation.

are used, which leads to the objective to be minimized for edge type r

$$\mathcal{O}^r = KL(W_{uv}^{(r)}, s_t(u,v)) = -\sum_{(u,v)\in\mathcal{P}^r} W_{uv}^{(r)} \log s_r(u,v) + const, \tag{6.2}$$

where $KL(\cdot)$ stands for the Kullback-Leibler divergence.

Further, substituting Eq. (6.1) into Eq. (6.2) and taking all edge types into account and, the overall objective function becomes

$$\mathcal{O} = -\sum_{\substack{(u,v)\in\mathcal{P}^r \\ r\in\mathcal{R}}} W_{uv}^{(r)} \log \frac{\exp(\boldsymbol{\mu}_r^\top \mathbf{g}_{uv})}{\sum_{\tilde{v}\in\mathcal{P}_{u*}^r} \exp(\boldsymbol{\mu}_r^\top \mathbf{g}_{u\tilde{v}}) + \sum_{\tilde{u}\in\mathcal{P}_{*v}^r} \exp(\boldsymbol{\mu}_r^\top \mathbf{g}_{\tilde{u}v})}. \tag{6.3}$$

To formulate edge embeddings required by Eq. (6.3), we derive from the same embeddings of the associated nodes regardless of the involved edge type, so that we reach a unified model where the learning process involving multiple edge types can work together and mutually enhance each other if they embody compatible semantics. While there are many options to build edge embedding from node embedding, we expect our formulation not to be over-complicated, so that the overall model could be computationally efficient and thereby easy-to-use. Moreover, in order for HEER to handle general HINs, it must also be able to handle directed and undirected edges accordingly. Considering these requirements, we decompose node embedding into two sections $\mathbf{f}_u = \begin{bmatrix} \mathbf{f}_u^O \\ \mathbf{f}_u^I \end{bmatrix}$, where $\mathbf{f}_u^O$ and $\mathbf{f}_u^I$ are two column vectors of the

same dimension, and build edge embedding on top of node embedding as

$$\mathbf{g}_{uv} \coloneqq \begin{cases} 2 \cdot \mathbf{f}_u^O \circ \mathbf{f}_v^I, & \text{directed representation from } u \text{ to } v, \\ \mathbf{f}_u^O \circ \mathbf{f}_v^O + \mathbf{f}_u^I \circ \mathbf{f}_v^I, & \text{undirected representation,} \end{cases} \tag{6.4}$$

where $\circ$ represents the Hadamard product. Besides Hadamard product, on can also build $\mathbf{g}_{uv}$ in a way similar to Eq. (6.4) using addition, subtraction, or outer-product. We leave the exploration of this direction to future works.

Taking Eq. (6.4) into account, learning node and edge embedding from an HIN by minimizing Eq. (6.3) is equivalent to the following optimization problem

$$\min_{\{\mathbf{f}_u\}_{u\in\mathcal{V}}, \{\boldsymbol{\mu}_r\}_{r\in\mathcal{R}}} \mathcal{O}. \tag{6.5}$$

6.4.2 Model Inference

The HEER model in Eq. (6.5) that we aim to infer can be structured as a neural network as illustrated in Figure 6.4, where $\boldsymbol{F} = \big[\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_{|\mathcal{V}|}\big] \in \mathbb{R}^{d_\mathcal{V} \times |\mathcal{V}|}$ and $\boldsymbol{M} = \big[\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_{|\mathcal{R}|}\big] \in \mathbb{R}^{d_\mathcal{E} \times |\mathcal{R}|}$. Each pair of nodes gets their respective embeddings through the dense layer $\boldsymbol{F}$, which further compose edge embedding by function $g(\cdot, \cdot)$. The raw scores for all edge types are obtained through another dense layer $\boldsymbol{M}$, followed by a type filter where the neuron for an edge type is connected to its corresponding neuron in the next layer only if this type is compatible with the node types of the input node pairs. Lastly, the loss is calculated by the typed closeness and the existence of edges in between the input node pair.

Since it is computationally expensive to compute the denominator in Eq. (6.1), we adopt the widely used negative sampling method [44], which enjoys linear-time computation. Specifically, each time, an edge between (u, v) with type r is sampled from the HIN with probability proportional to its weight. Then K negative node pairs $(u, \tilde{v}_i)$ and K negative node pairs $(\tilde{u}_i, v)$ are sampled, where each $\tilde{u}_i$ has the same type as u and each $\tilde{v}_i$ has the same type as v. The loss function computed from this sample becomes

$$\log \sigma(\boldsymbol{\mu}_r^\top \mathbf{g}_{uv}) + \sum_{i=1}^{K} \mathbb{E}_{\tilde{v}_i} \log \sigma(-\boldsymbol{\mu}_r^\top \mathbf{g}_{u\tilde{v}_i}) + \sum_{i=1}^{K} \mathbb{E}_{\tilde{u}_i} \log \sigma(-\boldsymbol{\mu}_r^\top \mathbf{g}_{\tilde{u}_i v}),$$

where $\sigma(\cdot)$ is the sigmoid function $\sigma(x) = \exp(x)/\big(1 + \exp(x)\big)$.

We adopt mini-batch gradient descent with the PyTorch implementation to minimize the loss function with negative sampling, where each mini-batch contains B sampled edges. We

also use the node embeddings pretrained by the homogeneous network embedding algorithm LINE [20] to initialize the node embeddings in HEER. The edge-type–specific scoring vector $\boldsymbol{\mu}_r$ is initialized to be all-one vectors.

6.5 EXPERIMENTS

In this section, we evaluate the embedding quality of the proposed method and analyze the utility of employing edge representation and heterogeneous metric using two large real-world HINs. We first perform an edge reconstruction task to directly quantify how well the embedding algorithms can preserve the information in the input HINs. Then, we conduct in-depth case studies to analyze the characteristics of the proposed method.

6.5.1 Baselines

We compare the proposed HEER algorithm with baseline methods that fit the setting of our problem, *i.e.*, the methods should be applicable to general HINs without the help of additional supervision or expertise.

- **Pretrained** (LINE [20]). This baseline uses the LINE algorithm to generate node embeddings, which are also used to initialize HEER. LINE is a homogeneous network embedding algorithm based on the skip-gram model [44]. We use inner product to compute the score of observing an edge between a pair of node embeddings following the original paper [20].

- **AspEm** [57]. AspEm is a heterogeneous network embedding method that captures the incompatibility in HINs by decomposing the input HIN into multiple aspects with an unsupervised measure using dataset-wide statistics. Embeddings are further learned independently for each aspect. This method considers the incompatibility in HINs but does not model different extent of incompatibility. Furthermore, it does not allow joint learning of embeddings across different aspects. Out of fairness, we let the number of aspects in AspEm to be two, in order to generate the final embedding with dimension that is identical to other methods. Inner product is also used to compute the score for this baseline.

- **UniMetrics** (metapath2vec++ [55]). This is a partial model of HEER, where the metrics $\{\boldsymbol{\mu}_r\}_{r\in\mathcal{R}}$ are not updated in the training process, *i.e.*, they remain uniform as initialized. It is equivalent to the metapath2vec++ [55] using all edges as length-1

meta-paths without further selection. This method restricts the negative sampling to be done within the consistent node types, *i.e.*, performs heterogeneous negative sampling, but still embeds all nodes into the same metric space regardless of types.

- **Pretrained + Logit**. On top of the embeddings from the previous Pretrained model, we train a logistic regression (Logit) model for each edge type using the input network. Then, we compute scores for test instances of each edge type using the corresponding Logit model. This method models heterogeneous metrics but does not allow the node embeddings and the edge embeddings to be further improved according to the inferred metrics.

6.5.2 Data Description and Experiment Setups

In this section, we describe the two real-world HINs used in our experiments as well as experiment setups.

Datasets. We use two publicly available real-world HIN datasets: DBLP and YAGO.

- **DBLP** is a bibliographical network in the computer science domain [118]. There are five types of nodes in the network: author, paper, key term, venue, and year. The key terms are extracted and released by Chen et al. [29]. The edge types include authorship (aut.), term usage (term), publishing venue (ven.), and publishing year (year) of a paper, and the reference relationship from a paper to another (ref.). We consider the first four edge types as undirected, and the last one as directed. The corresponding network schema is depicted in Figure 6.3b on page 76.

- **YAGO** is a large-scale knowledge graph derived from Wikipedia, WordNet, and GeoNames [119]. There are seven types of nodes in the network: person, location, organization, piece of work, prize, position, and event. A total of 24 edge types exist in the network, with five being directed and others being undirected. These edge types are illustrated together with the schema of the network in Figure 6.5.

We summarize the statistics of the datasets including the total number of nodes, the total number of edges, and the counts of each node type in Table 6.1.

Experiment Setups. For all experiments and all methods, we set the total embedding dimension to be 256. That is, for HEER and its related baselines, $\mathbf{f}_u \in \mathbb{R}^{256}$ and $\mathbf{f}_u^I, \mathbf{f}_u^O \in \mathbb{R}^{128}$, and each of the two aspects in AspEm uses a 128-dim embedding space. The pretrained model is always tuned to the best according to the performance in the edge reconstruction

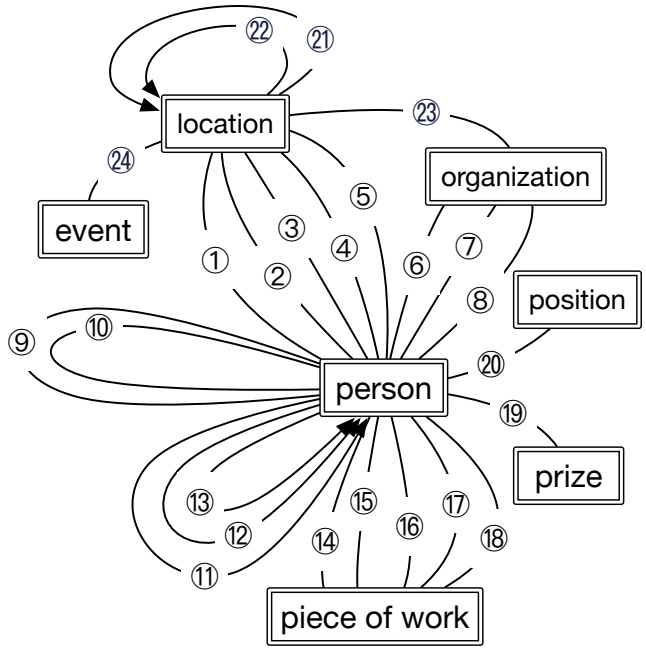


Figure 6.5: The schema of the YAGO network.

task to be introduced in Section 6.5.3. The negative sampling rate is always set to $K = 5$ for all applicable models. We always rescale the pretrained embedding by a constant factor of 0.1 before feeding them into HEER to improve the learning of heterogeneous metrics, which shares intuition with a previous study [120] in improving angular layout at the early stage of model training. The learning rate for gradient descent for HEER is set to 10 on both datasets. Note that we use the same set of hyperparameters for HEER on both DBLP and YAGO in order to provide an easy-to-use solution to the problem of comprehensive transcription of HINs without the hassle of extensive parameter tuning.

Table 6.1: Basic statistics for the DBLP and YAGO networks.

Dataset	Node	Edge	Node type	Edge type
DBLP	3,170,793	27,126,718	5	5
YAGO	579,721	2,191,464	7	24

Table 6.2: Per-edge-type, micro-average, and macro-average MRR achieved by each model in the edge reconstruction task.

Dataset	DBLP							YAGO	
Metric (MRR)	Aut.	Term	Ref.	Pub. venue	Pub. year	Micro-avg.	Macro-avg.	Micro-avg.	Macro-avg.
Pretrained (LINE [20])	0.7053	0.4830	0.8729	0.7488	0.4986	0.6307	0.6617	0.7454	0.6890
AspEm [57]	0.7068	0.6010	0.8648	0.7612	0.6791	0.6976	0.7225	0.7832	0.6825
UniMetrics (len-1 metapath2vec++ [55])	0.7040	0.5772	0.8466	0.7534	0.6781	0.6812	0.7119	0.7437	0.6884
Pretrained + Logit	0.8187	0.6996	0.8072	0.8379	0.4889	0.7310	0.7304	0.8233	0.7012
HEER	**0.8964**	**0.7188**	**0.9573**	**0.9132**	**0.7421**	**0.8189**	**0.8456**	**0.8635**	**0.7185**

6.5.3 Edge Reconstruction Experiment

In order to directly quantify the extent to which an embedding algorithm can preserve the information in the input HINs, we devise the edge reconstruction experiments for both datasets. For each HIN, we first knock out a portion of edges uniformly at random, with a certain knock-out rate $\kappa \in (0, 1)$. Embedding of the network after knock-out is then learned using each compared method. The task is to reconstruct the edges being knocked out using the learned embedding models.

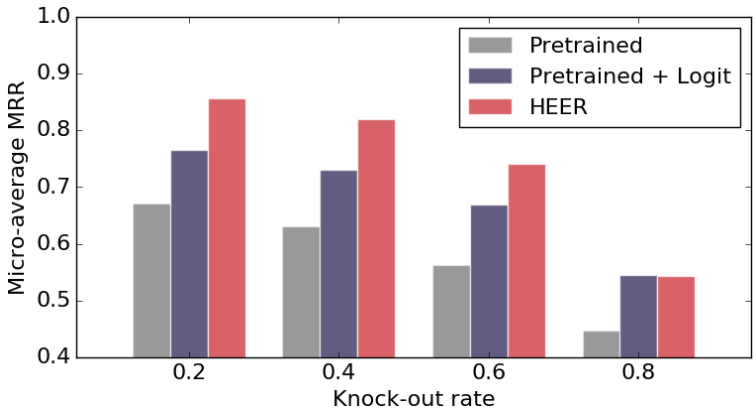
Specifically, for each edge that is knocked out from the network, suppose it is between node pair (u, v) and of edge type r, we randomly sample 10 negative pairs $(u, \tilde{v})$ that do not have type-r edges in the original full network, where $\tilde{v}$ is of the same node type as v. For any model after training, a score can be calculated to reflect the likelihood for each of the 11 node pairs to be associated by type-r edge in the current model. The reciprocal rank is then computed to measure the quality of the model, where the reciprocal rank is the reciprocal of the rank of the positive pair among all 11 node. Similarly, another reciprocal rank is computed for the same node pair (u, v) and 10 other randomly sampled negative pairs $(\tilde{u}, v)$ with fixed v but sampled $\tilde{u}$. Finally, we report the mean reciprocal rank (MRR), which is computed by the mean of reciprocal ranks for the target test instances. In particular, the micro-average MRR and the macro-average MRR are reported for both DBLP and YAGO, where the micro-average MRR is computed by the mean of all reciprocal ranks computed regardless of edge types, while the macro-average MRR is derived by first computing the mean of reciprocal ranks for each edge type, and then averaging all these means across different edge types. Additionally, we also report the MRR for each edge type for DBLP, since DBLP involves only 5 edge types, while YAGO has as many as 24 edge types. We present the results with knock-out rate $\kappa = 0.4$ in Table 6.2.

Modeling incompatibility benefits embedding quality. As shown in Table 6.2, the proposed HEER model outperformed all baselines in both datasets under both micro-average MRR and macro-average MRR, which demonstrated the effectiveness of the proposed method. Even when looking at each edge type in DBLP, the MRR achieved by HEER was still the best. Besides, in DBLP, AspEm outperformed Pretrained and UniMetrics on most metrics. Recall that AspEm decomposed the HIN into distinct aspects using dataset-wide statistics. As a result, it forbade semantically incompatible edge types to negatively affect each other in the embedding learning process and thereby achieved better results. In YAGO, the baselines considering heterogeneity did not always clearly outperform the simplest baseline Pretrained. We interpret this result by that YAGO has much more edge types than DBLP, which introduces even more varied extent of incompatibility, and the relatively simple approaches adopted by AspEm and UniMetrics in modeling incompatibility may not be enough to bring in significant performance boost. In contrast, armed with heterogeneous metrics fine-grained to the edge type level, HEER outperformed Pretrained by a clear margin even in YAGO.

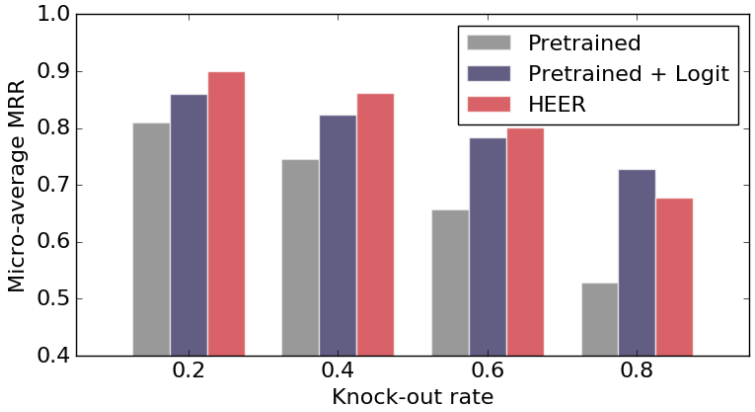
Heterogeneous metrics helps improving embedding quality. As a sanity check, the Pretrained + Logit model helps rule out the possibility that HEER archives better results only by learning edge-type–specific metrics without actually improving embedding quality. From Table 6.2, it can be observed that by coupling with the additional edge-type–specific logistic regression and modeling heterogeneous metrics, the performance was improved on top of the Pretrained mode. This observation further consolidated the necessity of employing heterogeneous metrics for different edge types in solving the problem of comprehensive transcription of heterogeneous information networks. However, Pretrained + Logit still performed worse than the proposed HEER mode, which implies that the inferred heterogeneous metrics of HEER indeed in return improved the quality of the node and edge embedding.

Heterogeneous negative sampling is not always enough to capture incompatibility. UniMetric performed better than the Pretained model in DBLP, it failed to make an absolute win over Pretained as other methods did in the YAGO dataset. Our interpretation of this result is that while the heterogeneous negative sampling as used by UniMetric does leverage certain type-specific information in HINs, it may not be always enough to model incompatibility and resolve the negative impact it brings to embedding quality. This observation is to be further corroborated in Section 6.5.4 by examining the capability of transcribing information implied by meta-paths in each model.

Varying Knock-Out Rate in Edge Reconstruction. Additionally, we vary the knock-out rate κ on both datasets and report the micro-average MRR for the proposed HEER



(a) DBLP.



(b) YAGO.

Figure 6.6: Micro-average MRR under multiple knock-out rate κ in the egde reconstruction tasks.

model and two baseline models that require less training time. As presented in Figure 6.6, HEER outperformed all baselines under at most knock-out rates, which demonstrated the robustness of the proposed model. Besides, Pretrained + Logit outperformed Pretrained at all knock-out rates, which is also in line with the previous results we have presented. Notably, HEER did not outperform Pretrained + Logit when the knock-out rate $\kappa = 0.8$. This is explainable because only a very small portion (20%) of the original HIN was used for learning embedding when $\kappa = 0.8$. With a bigger model size than Pretrained + Logit, HEER was more prone to suffering from over-fitting.

6.5.4 Case Studies

In this section, we conduct a series of in-depth case studies to understand the characteristics of the proposed HEER model.

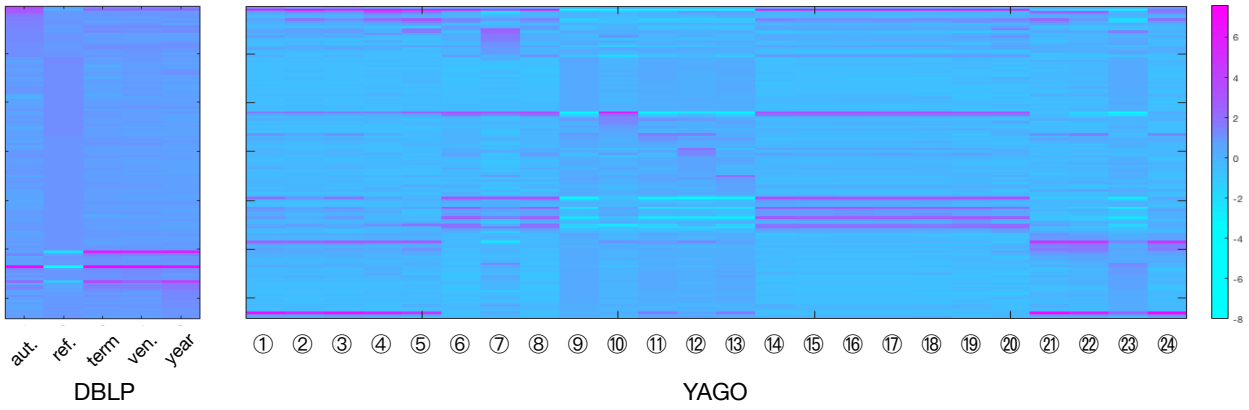


Figure 6.7: The learned heterogeneous metrics of the HEER model sensed the heterogeneous semantics of the HINs.

Learned heterogeneous metrics. HEER leverages heterogeneous metrics to model the different extent of incompatible semantics carried by different edge types. In this section, we analyzed the learned metrics $\{\boldsymbol{\mu}_r\}_{r\in\mathcal{R}}$ in HEER to verify if they indeed captured the different semantics and thereby enriched the model capability.

To this end, we use heat maps to illustrate $\{\boldsymbol{\mu}_r\}_{r\in\mathcal{R}}$ that are learned in the edge reconstruction experiments on both HINs. Specifically, for each dataset, we first standardize the elements of each $\boldsymbol{\mu}_r$ to have zero mean and unit deviation, so that $\boldsymbol{\mu}_r$'s have comparable scales after standardization for all $r \in \mathcal{R}$. Then, we re-order the $d_{\mathcal{E}}$ dimensions for better visualization and plot the heat maps in Figure 6.7.

Recall that the inferred metrics $\{\boldsymbol{\mu}_r\}_{r\in\mathcal{R}}$ were set to be all-one vectors in initialization, whereas Figure 6.7 shows that different metrics have generally reached different distributions over the $d_{\mathcal{E}}$ dimensions after training. This implies that the inferred metrics of the HEER model indeed sensed the heterogeneity of the HINs, and were using different projected metric spaces to embed different semantics of the input network. Notably, it can be seen from the heat map of YAGO that edge types ⑥ (isAffiliatedTo) and ⑧ (playsFor) have similar inferred metrics. This is actually expected because these edge types are often associated with the relationship between professional sports players and their associated teams in YAGO. Besides, similar phenomenon can be observed between ⑨ (isMarriedTo) and edge type ⑪ (hasChild).

Embedded subnetwork with different edge types. In order to understand how the network is impacted by the introduction of heterogeneous metrics, we took a closer look at a subnetwork surrounding the British writer Thomas Hardy in the YAGO dataset. Multiple other writers having the *influences* relationship (⑫, colored gray) with Thomas Hardy include Charles Dickens, John Fowles, and John Irving, while Florence Dugdale and Thomas

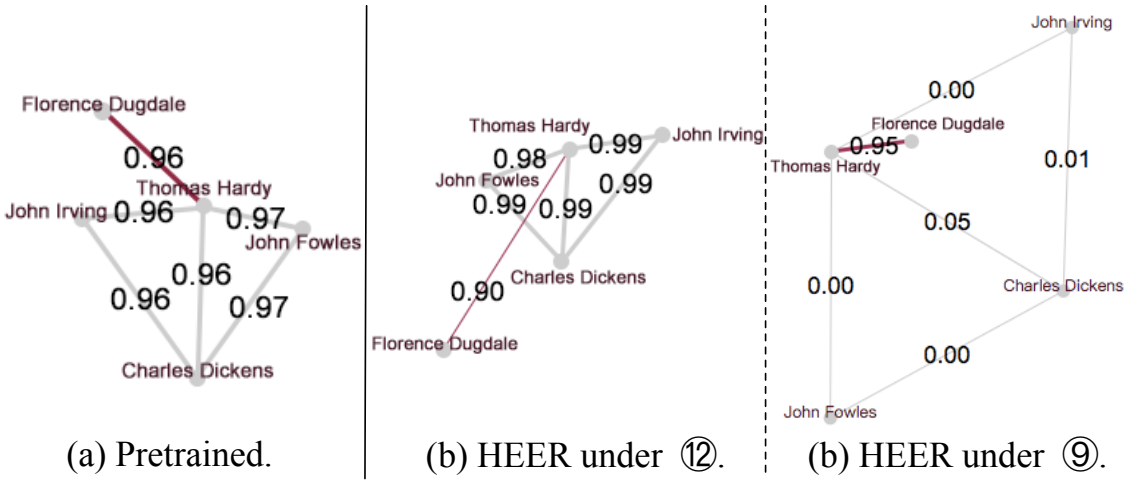


Figure 6.8: The subnetwork surrounding Thomas Hardy in YAGO in multiple embedding models under potentially different metrics.

Hardy enjoyed the *isMarriedTo* relationship (⑨, colored red). Besides, Fowles and Irving are also influenced by Dickens. In Figure 6.8, we visualized this subnetwork under each embedding model with the inferred possibility of edge existence marked, where the embedding models are training using the entire network. It can be seen from Figure 6.8a that without distinguishing edge types, Pretrained assigned high possibilities to all edges. Meanwhile, with the learned heterogeneous metrics, the HEER model assigned a relatively low probability for Dugdale and Hardy under the metric for *influences* as in Figure 6.8b (note that Dugdale was also a writer), and a clearly higher probability under the metric for *isMarriedTo* as in Figure 6.8c.

6.5.5 Efficiency Study

For efficiency study, we plot out the loss and the performance of the proposed HEER algorithm against the number of epochs in the edge reconstruction experiment. We also illustrate the same curves of the UniMetrics (len-1 metapath2vec++) model for comparison. The results are presented in Figure 6.9.

Judging from the curve for the loss again the number of epochs in Figure 6.9, HEER converges at a comparable rate with the skip-gram–based UniMetrics (metapath2vec++). Besides, HEER took less than twice as much time to finish each epoch as metapath2vec++ did. This is expected because HEER only additionally requires one-step gradient descent for one $\boldsymbol{\mu}_r$ when training on each sampled training example. As a result, the time complexity of HEER for each epoch differs from that of metapath2vec++ by a small constant factor.

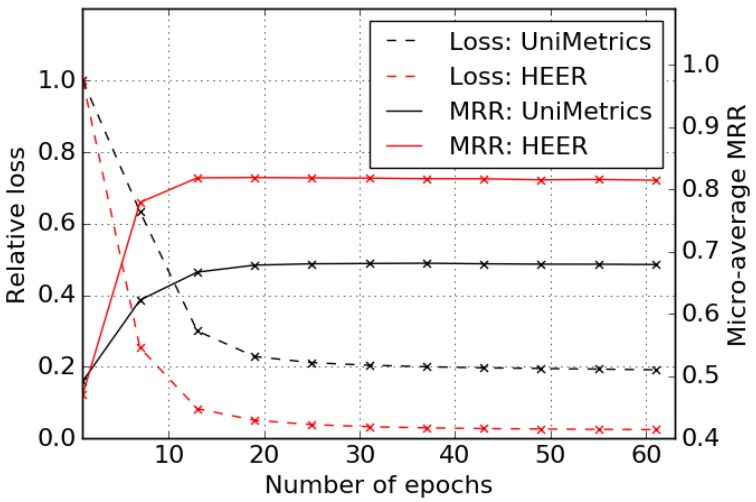


Figure 6.9: The relative loss and the micro-average MRR against the number of epochs for the proposed HEER model and the UniMetrics (len-1 metapath2vec) model.

Combining the above two properties, HEER enjoys overall complexity linear to the number of nodes as skip-gram–based algorithms do [18, 49, 55].

6.6 SUMMARY

We studied the problem of the comprehensive transcription of HINs in embedding learning, which preserves the rich information in HINs and provides an easy-to-use approach to unleash the power of HINs. To tackle this problem, we identify that different extents of semantic incompatibility exist in real-world HINs, which pose challenges to the comprehensive transcription of HINs. To cope with these challenges, we propose an algorithm, HEER, that leverages edge representations and heterogeneous metrics. Experiments and in-depth case studies with large real-world datasets demonstrate the effectiveness of HEER and the utility of edge representations and heterogeneous metrics.

CHAPTER 7: EXPLOITING HETEROGENEOUS ASSOCIATION IN HYPERNYMY DISCOVERY FROM NETWORKS

7.1 OVERVIEW

Heterogeneous information network (HIN) has been heavily studied since the past decade [2, 14, 15, 16]. Many real-world HINs are essentially text-rich since some of their nodes are associated with additional textual information [34, 121, 122, 123, 124]. For example, the nodes representing research papers in the bibliographical network, DBLP, are naturally associated with their textual contents. Such text-rich HINs is capable of encapsulating rich information and has been shown to be useful in tasks such as clustering [123], topic modeling [121, 122], and embedding learning [34].

In the meantime, hypernymy discovery is an important and fundamental problem [67, 125, 92]. Hypernymy, also know as type-of relation or lexical entailment relation [126], refers to the relation between a hypernym and a hyponym, where a term t_1 is called a hyponym of a term t_2 and t_2 is called a hypernym of t_1 if t_1 can be categorized under t_2. We denote this relation by $t_1 \rightarrow t_2$. For instance, "animal" is a hypernym of "bird", and "machine learning algorithm" is a hyponym of "computer science". Being able to discover quality hypernymy can benefit downstream applications such as taxonomy construction and question answering [125, 92]. For example, algorithms have been proposed to construct taxonomy given a list of hypernymy pairs [83, 127]. Even in the case where the discovered hypernymy pairs are imperfect, these pairs can still largely facilitate the process of the human curation by recommending hypernyms and hyponyms to expand the taxonomy under construction.

Existing hypernymy discovery methods mainly detect hypernymy pairs from large corpora with hypernyms and hyponyms being terms from a given vocabulary $\mathcal{D}$. We instead propose to tackle the problem by discovering hypernymy from text-rich HINs. Particularly, in the DBLP network with node type *paper*, *author*, *publishing venue*, *keyword*, *etc.*, we may discover hypernymy among all *keyword* nodes, and in a professional social network with node type *user*, *employer*, *school*, *skill*, *etc.*, we may discovery hypernymy among all skills. The intuition is that while the textual part of a text-rich HIN constitutes a corpus from which existing methods can mine hypernymy, the network part of the text-rich HIN can provide additional high-quality signals. In fact, a major line of methods for discovering hypernymy relies on the co-occurrence of terms in the input corpus [67, 68]. Such textual co-occurrence can be noisy, while the relationships represented by edges and paths in an HIN are more concrete. For example, in the DBLP network, if two authors are linked to the same paper, we would know they are co-authors, whereas we cannot safely make the same judgment simply

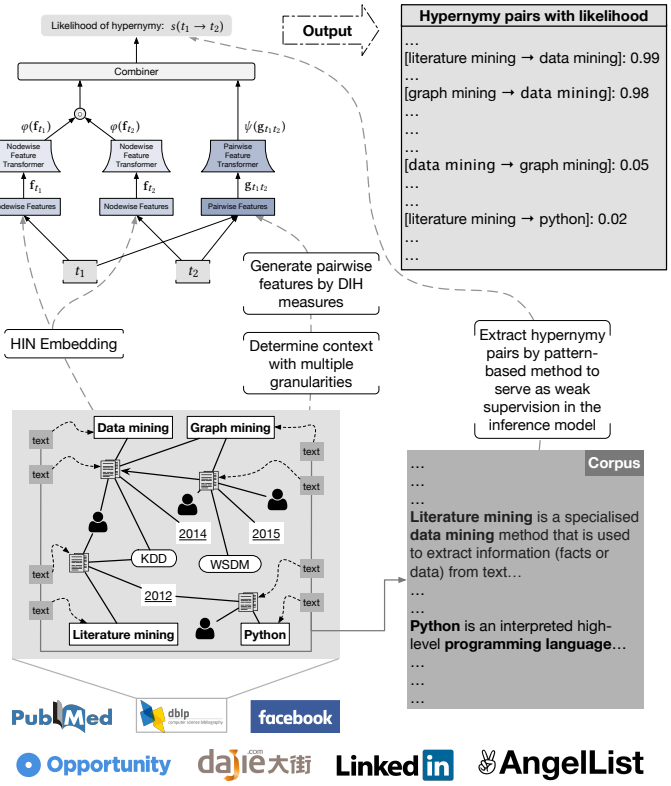
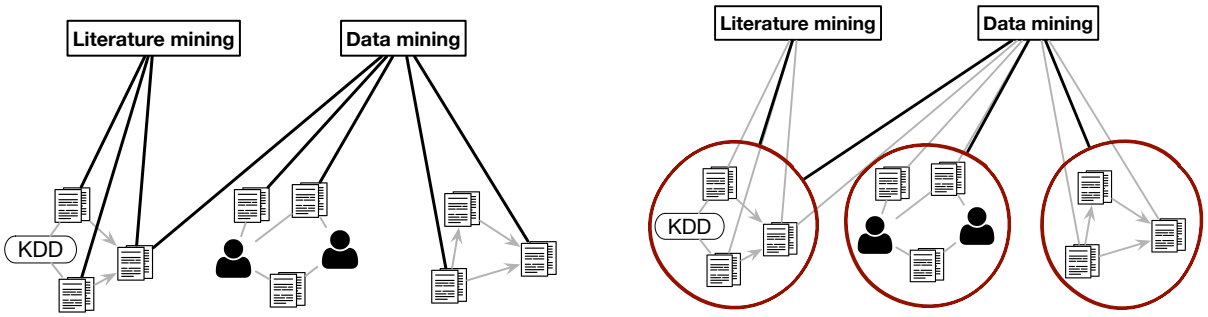


Figure 7.1: The overview of the proposed HDCG framework for unsupervised hypernymy discovery from one of the many text-rich HINs. We discover hypernymy by exploiting additional rich signals from the network data besides the corpus. A neural network model is used to discover hypernymy, which consumes weak supervision with relatively high precision extracted from the textual part of the text-rich HIN by our framework. A rich pool of features with comparably good term pair coverage is generated from the network part. A close-up illustration of the neural network is to be provided in Figure 7.5.

because they often co-occur in a corpus. The formal definition of the problem of hypernymy discovery from text-rich HINs will be provided in Section 7.2.

Unsupervised methods for hypernymy discovery from corpus typically fall into two categories: the textual-pattern–based methods [80, 82] and the distributional methods [74, 72]. While the textual-pattern–based methods can only be applied to the textual part of a text-rich HIN, the distributional methods can be extended to decode the signals from network data. Most distributional methods are developed based on the distributional inclusion hypothesis (DIH) [67]. The DIH assumes that the context of a hypernym always subsumes the context of a hyponym, and we will formally define the concept of context in Section 7.2. That is, if we denote $\mathcal{C}^{(t)}$ the set of all units in the context that are relevant to term $t \in \mathcal{D}$, the DIH would assert $\mathcal{C}^{(t_1)} \subseteq \mathcal{C}^{(t_2)}$ if t_1 is a hyponym of t_2. If we were to discover hypernymy



(a) A simple difinition of context where each paper is a contextual unit.

(b) A definition with more coarse granularity where each red circle is a contextual unit.

Figure 7.2: Example of the two contexts with different granularities.

pairs among keywords in the DBLP network, the simplest way to define the context $\mathcal{C}$ is to let each paper node be a contextual unit, since *paper* is the only node type directly linked to *keyword* as to be illustrated later in Figure 7.3a.

However, such a simple definition of context may not be proper for hypernymy discovery. Figure 7.2 gives a hypothetical example with two keywords "literature mining" (t_1) and "data mining" (t_2). Under the above simple definition, all papers linked to "literature mining" constitutes $\mathcal{C}^{(t_1)}$ and those linked to "data mining" forms $\mathcal{C}^{(t_2)}$. Since "data mining" is a hypernym of "literature mining", if the DIH holds in this context, we should have $\mathcal{C}^{(t_1)} \subseteq \mathcal{C}^{(t_2)}$. However, a paper with "literature mining" linked as a keyword may not need to additionally tag the more general "data mining", and such scenario as illustrated in Figure 7.2a would violate the assumption of the DIH. This problem can happen whenever the hypernym is too general, and the contextual units are too fine-grained. Fortunately, this issue can be resolved if we redefine the context $\mathcal{C}$ so that each new contextual unit is a group of semantically relevant papers instead of individual papers. Under this new definition of $\mathcal{C}$, the desired property $\mathcal{C}^{(t_1)} \subseteq \mathcal{C}^{(t_2)}$ would hold as shown in Figure 7.2b. The contextual units redefined as such have more coarse granularity. In other words, they are conceptually more general, and each of them is relevant to more terms from the target vocabulary $\mathcal{D}$. In fact, we observe that hypernymy should be revealed at multiple granularities, since the generality of a hypernymy pair is coupled with the granularity of the context, and the availability of HIN data enables us to solve the problem in a broad spectrum of context granularity. We shall further illustrate this point using examples from real-world data in Section 7.3.

Additional challenges for the problem of hypernymy discovery from text-rich HINs lie in how to combine signals from the textual part and the network part of the input data. To tackle this challenge we extract relatively high-precision hypernymy pairs from the text part

and serve them to a neural network as weak supervision, while deriving features with high recall from the network part. We refer to this proposed framework as HDCG – short for **h**ypernymy **d**iscovery featuring **c**ontext **g**ranularity.

Lastly, we summarize our contributions as follows:

1. We propose to discover hypernymy from text-rich heterogeneous information networks (HINs), which introduces additional high-quality signals beyond textual corpora.

2. We identify the impact of context granularity on the distributional inclusion hypothesis (DIH).

3. We propose the HDCG framework exploiting both network and textual data in text-rich HINs, which also enjoys a more informative DIH features by modeling context granularity.

4. Experiments validated the utility of modeling context granularity and the effectiveness of leveraging additional HIN signals in hypernymy discovery. A case study on a downstream application showed that a reasonable taxonomy could be generated using hypernymy discovered by HDCG.

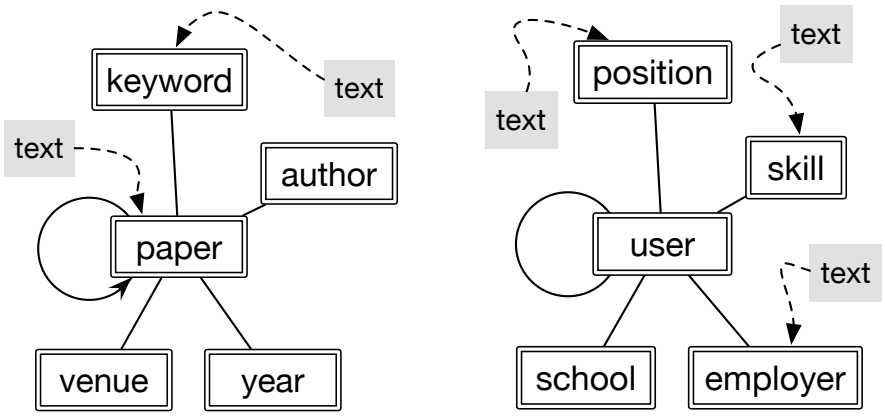
7.2 PRELIMINARIES

We define related concepts and notations in this section.

Definition 7.1 (Text-Rich Heterogeneous Information Network [14])**.** *An* ***information network*** *is a directed graph $G = (\mathcal{V}, \mathcal{E})$ with a node type mapping $\varphi : \mathcal{V} \to \mathcal{T}$ and an edge type mapping $\psi : \mathcal{E} \to \mathcal{R}$. When $|\mathcal{T}| > 1$ or $|\mathcal{R}| > 1$, the network is called a* ***heterogeneous information network*** *(HIN). An HIN is referred to as a* ***text-rich HIN*** *if a portion of its nodes are associated with textual information that collectively constitute a* ***corpus*** *−.*

Given the typed essence of HINs, the network schema $\tilde{G} = (\mathcal{T}, \mathcal{R})$ [14] is used to abstract the meta-information regarding the node types and edge types in an HIN. Figure 7.3a illustrates the schema of the DBLP network, where a *paper* node may have additional textual information from its content and a *keyword* node may be associated with its description from Wikipedia. Similarly, the schema of a social network in Figure 7.3b consists of 5 node types with *skill*, *employer*, and *position* having textual information.

Definition 7.2 (Hypernymy Discovery from Text-Rich Heterogeneous Information Networks)**.** *Given a text-rich HIN $G = (\mathcal{V}, \mathcal{E})$ and a* ***target node type*** *where each node of*



(a) DBLP. **(b) A professional social network (PSN).**

Figure 7.3: The schemata of two text-rich HINs.

the target type must correspond to a textual term, we refer to the set of all such terms as the ***target vocabulary*** $\mathcal{D}$*. The problem of* ***hypernymy discovery from this text-rich HIN*** *aims to discover a list of hypernymy pairs with confidence scores where the hypernyms and the hyponyms are terms from* $\mathcal{D}$*.*

In the heterogeneous bibliographic network, DBLP, the target node type can be *keyword*, and in a heterogeneous professional social network, the target node type can be *skill*.

Definition 7.3 (Context in DIH Measures)**.** *Measures based on the distributional inclusion hypothesis are defined on a given domain of* ***context*** $\mathcal{C}$*, over which each term in the target vocabulary* $\mathcal{D}$ *has a relevance distribution. For a term* $t \in \mathcal{D}$ *and a* ***contextual unit*** $c \in \mathcal{C}$*, denote* $r_c(t)$ *the relevance between* t *and* c*.*

Additionally, we denote the subdomain of the context that are relevant to term t by $\mathcal{C}^{(t)} \coloneqq \{c \in \mathcal{C} \,|\, r_c(t) \neq 0\}$. The primary intuition of measures based on the distributional inclusion hypothesis is that $\mathcal{C}^{(t_2)}$ should include $\mathcal{C}^{(t_1)}$ if t_2 is a hypernym of t_1, and one widely-used DIH measure, *WeedsPrec* [72], is given by

$$M_1(t_1 \to t_2) = \frac{\sum_{c \in \mathcal{C}^{(t_1)} \cap \mathcal{C}^{(t_2)}} r_c(t_1)}{\sum_{\mathcal{C}^{(t_1)}} r_c(t_1)}. \tag{7.1}$$

The higher the *WeedsPrec* score, the more likely t_2 is a hypernym of t_1. In the traditional task of hypernymy discovery from a corpus, the typical definition of C is the set of all contextual terms in the corpus [67, 69, 72]. A term t and a contextual term c will have non-zero relevance if they co-occur.

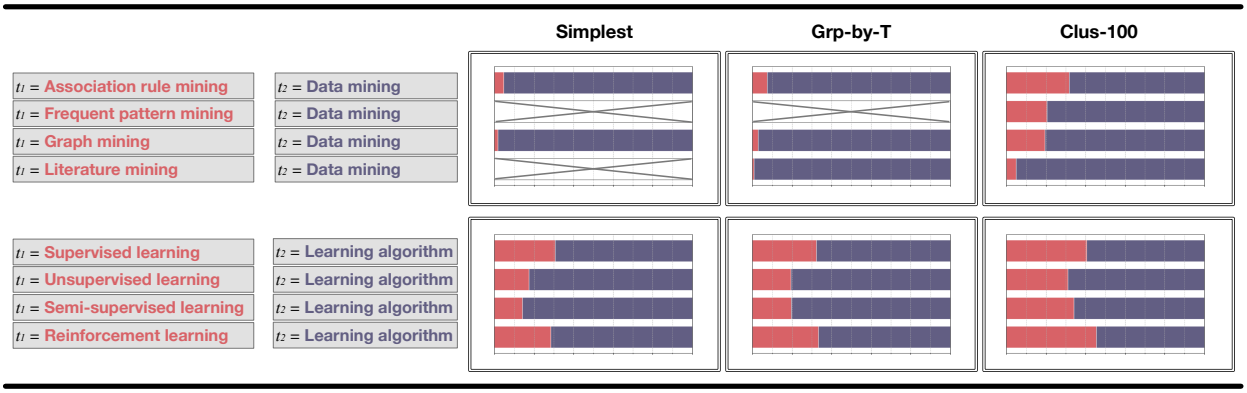


Figure 7.4: Using the DBLP dataset, we demonstrate the observation that different hypernymy pairs should be discovered at different context granularity. Each row in the plot corresponds to a hypernymy pair $t_1 \rightarrow t_2$, and each column corresponds to a context granularity. In each bar, the length ratio between the red part on the left and the blue part on the right is proportional to the ratio between $M_1(t_2 \rightarrow t_1)$ and $M_1(t_1 \rightarrow t_2)$.

7.3 CONTEXT GRANULARITY IN REAL-WORLD DATASET

In this section, we further illustrate the observation that different hypernymy pairs should be discovered at different context granularity using a real-world dataset. Figure 7.4 presents the *WeedsPrec* scores (M_1) between eight hypernymy pairs computed at three contexts with different granularities – *Simplest*, *Grp-by-T*, and *Clus-100* – and we will introduce their concrete definitions in Section 7.4.1. For each bar, the blue part on the right end is expected to be clearly longer than the red part on the left end, so that *WeedsPrec* may be able to reveal the hypernymy $t_1 \rightarrow t_2$ instead of equivocally predicting the reverse $t_2 \rightarrow t_1$ could also be true.

In the first column (*Simplest*) of the figure, the ratio between $M_1(t_1 \rightarrow t_2)$ and $M_1(t_2 \rightarrow t_1)$ cannot be visualized for two pairs involving "data mining" since their M_1 are trivially zero. This undesirable result is the outcome of the fact that "frequent pattern mining" and "data mining" are never linked to the same contextual unit in the DBLP dataset to be described in Section 7.5.1, and likewise for "literature mining" and "data mining". In the context represented by the second column (*Grp-by-T*), *WeedsPrec* is still trivially zero for one pair. Fortunately, if we choose the context in the last column, *WeedsPrec* can generate scores for all four pairs with hypernymy "data mining". However, the context in the last column (*Clus-100*) is not proper for all hypernymy pairs either. In the example of "reinforcement learning" (t_1) and "learning algorithm" (t_2), $M_1(t_1 \rightarrow t_2)$ and $M_1(t_2 \rightarrow t_1)$ are close to each other, which makes it hard to decisively assert "reinforcement learning" is a hyponym

of "learning algorithm". On the other hand, the distinction between $M_1(t_1 \rightarrow t_2)$ and $M_1(t_2 \rightarrow t_1)$ are much wider at the *Simplest* context in the first column. We interpret this result as the generality of a hypernymy pair is coupled with the granularity of the context, and hypernymy relations should, therefore, be revealed at multiple granularities.

Lastly, resolving the problem of the DIH shown in Figure 7.2a and Figure 7.4 by exploiting context granularity is easier with the availability of HINs as input, because in an HIN, one can easily define semantically meaningful contextual units using explicit network structures such as grouping by a specific node type or more complex structures including meta-paths or motifs [2, 14]. One may also use network clustering methods to derive contextual units on a broad spectrum of granularity by varying the number of clusters.

7.4 THE HDCG FRAMEWORK

We tackle the problem of discovering hypernymy from text-rich HINs by marrying the signals from the network part and the text part with a neural network model, which infers the likelihood of a term $t_1 \in \mathcal{D}$ being a hyponym of another $t_2 \in \mathcal{D}$.

The intuition is that, as we have reviewed in Section 2.5, pattern-based methods tend to achieve high precision with compromised recall [67, 81, 82, 84, 85], we extract hypernymy pairs from the corpus of a text-rich HIN and serve them as weak supervision to the model to be proposed. In order to achieve good recall, we generate a rich pool of features from the HINs. Particularly, we decode the network signals into nodewise features by HIN embedding and into pairwise features using the DIH-based measures under various context granularities.

Note that while the component of the neural network model is weakly supervised, the whole HDCG framework is unsupervised.

7.4.1 Nodewise Feature Generation

Network embedding has emerged as a powerful representation learning approach, which has been proven effective in many application scenarios [128]. A network embedding algorithm generally learns an embedding function $f : \mathcal{V} \rightarrow \mathbb{R}^d$ that maps a node to a d-dimensional vectorized representation. In our framework, we generate a feature $\mathbf{f}_v \coloneqq f(v)$ for each node $v \in \mathcal{V}$ using a network embedding algorithm designed for HINs from an existing study [129]. As such, each term t in the target vocabulary $\mathcal{D} \subseteq \mathcal{V}$ also attains a nodewise feature $\mathbf{f}_t$.

7.4.2 Pairwise Feature Generation

As introduced in Section 7.1 and 7.3, the DIH measures can be naturally extended to generate pairwise features from networks, whose power can be further unleashed by modeling context granularity. For each term pair $(t_1, t_2) \in \mathcal{D} \times \mathcal{D}$, we generate a feature using one of the many DIH measures under one specific context definition. Each such DIH measure in our framework is henceforth referred to as a base DIH measure. We introduce the base DIH measures used in the proposed framework together with the approaches to define a context with different granularities as follows.

Base DIH measures. For given context $\mathcal{C}$, we use the following four DIH measures.

- **M1** (*WeedsPrec* [72]) is one of the pioneering DIH measures. $M_1(t_1 \to t_2) = \sum_{c \in \mathcal{C}^{(t_1)} \cap \mathcal{C}^{(t_2)}} r_c(t_1) / \sum_{c \in \mathcal{C}}$

- **M2** (*invCL* [75]) is another widely used DIH measure which considers not only how likely t_1 is a hyponym of t_2 but also how unlikely t_1 is not a hypernym of t_2. It is defined by $M_2(t_1 \to t_2) = \sqrt{\mathit{ClarkDE}(t_1 \to t_2) \cdot [1 - \mathit{ClarkDE}(t_2 \to t_1]}$, where $\mathit{ClarkDE}(t_1 \to t_2) = \sum_{c \in \mathcal{C}^{(t_1)} \cap \mathcal{C}^{(t_2)}} \min(r_c(t_1), r_c(t_2)) / \sum_{c \in \mathcal{C}^{(t_1)}} r_c(t_1)$.

- **M3** shares the intuition of **M2**. $M_3(t_1 \to t_2) = \mathit{ClarkDE}(t_1 \to t_2) - \mathit{ClarkDE}(t_2 \to t_1)$.

- **M4**. A symmetric distributional measure, which technically does not capture the inclusion intuition of the DIH. We use it to quantify the relevance of the term pair. $M_4(t_1 \to t_2) = \sum_{c \in \mathcal{C}^{(t_1)} \cap \mathcal{C}^{(t_2)}} \min(r_c(t_1), r_c(t_2)) / |\mathcal{C}|$.

Determination of context. As discussed in Section 7.1, the simplest way to define context given an HIN and a target node type is to let every node linked to nodes of the target type be a contextual unit. We call the context $\mathcal{C}$ defined in this way the *Simplest* context. Also discussed in Section 7.1, atop the *Simplest*, one may redefine contextual units by grouping the original ones that are semantically relevant. With the availability of HIN data, we adopt the following two approaches to alternatively define the context in a broad spectrum of context granularity.

- **Define the context by explicit network structures.** Many explicit network structures can be found in HINs such as the node types, the edge types, the meta-paths, and the meta-graphs [14, 2]. Using these structures, one can design methods to group the original contextual units in the *Simplest* together to derive new contextual units. In this dissertation, we adopt the most straightforward way and group together all contextual units linked to nodes of a specific node type. We refer to this approach

as *Grp-by-type* with *type* being a specific node type. As an example, in the DBLP dataset, a contextual unit in *Grp-by-author* is the collection of all papers written by a particular author. We consider a term $t \in \mathcal{D}$ relevant to a contextual unit in *Grp-by-type* as long as t is relevant to at least one original unit that is grouped into the new unit.

- **Define the context by network clustering.** Another way to derive semantically meaningful groups is by network clustering. A great many clustering algorithms have been proposed for clustering HINs [14, 2]. With an intention to experiment with a simple algorithm while leveraging the rich information from HINs, we perform the classic K-means algorithm on the node features $\mathbf{f}_v \in \mathbb{R}^d$ to derive K clusters. Similarly, a term $t \in \mathcal{D}$ is relevant to a cluster-based contextual unit as long as t is relevant to at least one original unit in this cluster. This approach is henceforth referred to as *Clus-K*.

We remark that the above two approaches both yield contextual units with granularity coarser than the *Simplest*, while one can also define context granularity finer than the *Simplest*. For instance, using an explicit network structure, meta-graph [2], a definition of a finer contextual unit in the DBLP network can be two papers written by the same author. Under this definition, only two keywords simultaneously tagged to an authors' two papers are considered linked to a common contextual unit. Besides, we focus our investigation on the benefit of introducing HIN signals and the utility of modeling context granularity, and we hence always set the relevance to be binary, *i.e.*, $r_c(t) = 1$ if relevant and 0, otherwise.

For each term pair $(t_1, t_2) \in \mathcal{D} \times \mathcal{D}$, we compute one score using each one of the base DIH measures under each one of the contexts, which together constitute the pairwise feature $\mathbf{g}_{t_1 t_2}$ for (t_1, t_2). In other words, the dimension of $\mathbf{g}_{t_1 t_2}$ equals to the number of DIH-based measures (4) times the number of contexts.

Note that in practice, for any choice of context with too many non-zero $r_c(t)$, one may downsample the data by randomly dropping a portion of contextual units – rather than randomly setting a portion of non-zero $r_c(t)$ to zero – in the hope of reducing the change in score derived from DIH measures.

7.4.3 Weak Supervision from Text-Rich HIN

We generate weak supervision for the hypernymy inference model from the corpus − of the input text-rich HIN. As a pioneering method, the Hearst patterns [80] have been shown to have decent precision [67, 81, 82, 84, 85]. We use this method to extract a list $\mathcal{S} =$

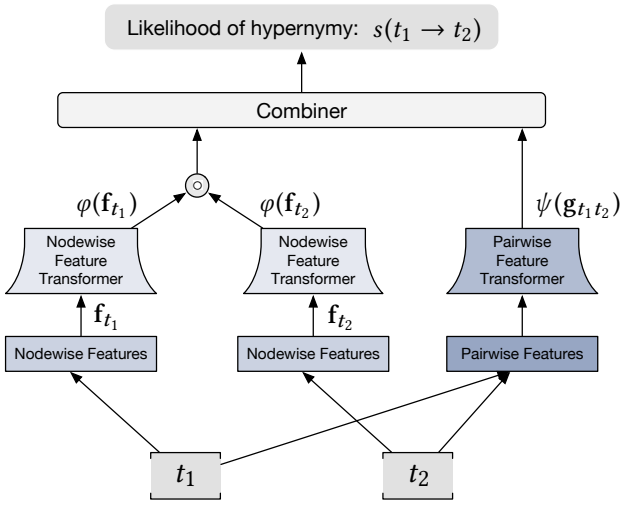


Figure 7.5: The proposed hypernymy inference model.

$\{(t_i^{\curlyvee}, t_i^{\curlywedge})\}_{i=1}^{|\mathcal{S}|}$ of hypernymy pairs from the corpus $-$. In Section 7.5, we shall quantitatively evaluate the validity of this method for generating weak supervision.

7.4.4 Hypernymy Inference Model

We aim to obtain a model that calculates the likelihood whether a pair of terms (t_1, t_2) are hyponym and hypernym using weak supervision extracted from the text part and features from the network part of the input text-rich HIN. The architecture of our hypernymy inference model with three major components is depicted in Figure 7.5. The first component is a nodewise feature transformer $\varphi(\cdot)$ that takes the raw nodewise feature $\mathbf{f}$ as input and transforms it into a new embedding space where the hypernymy semantics can be better captured. We design this transformer to be a simple linear layer with dropout followed by a non-linear activation using $tanh(\cdot)$ function. Following the core idea of the Siamese Network [130, 131], we apply the same nodewise feature transformer to both term t_1 and t_2. The second component is a pairwise feature transformer $\psi(\cdot)$ that acts upon the DIH-based pairwise features. Similarly, we design the pairwise feature transformer using a fully connected neural network with two hidden layers of size N and $N/2$, where N is the dimension of $\mathbf{g}_{t_1 t_2}$. Again, we apply dropout for regularization and use $tanh(\cdot)$ function for activation. The third component is a combiner that aggregates both nodewise and pairwise features after transformation and calculates the hypernymy score by

$$s(t_1 \rightarrow t_2) = \varphi(\mathbf{f}_{t_1})^T \mathbf{\Sigma} \varphi(\mathbf{f}_{t_2}) + \mathbf{h}\psi(\mathbf{g}_{t_1 t_2}), \tag{7.2}$$

where $\mathbf{\Sigma}$ is a diagonal matrix and $\mathbf{h}$ is a column vector.

To learn the parameters in both $\mathbf{\Sigma}$ and $\mathbf{h}$, we expect hypernymy pairs to have greater hypernymy scores than non-hypernymy pairs. Therefore, we use the following contrastive loss for model learning

$$\mathcal{L} = \sum_{(t^{\curlyvee}, t^{\curlywedge}) \in \mathcal{S}} \sum_{t^{\times} \in \mathcal{N}(t^{\curlywedge})} \max\left[0, 1 - s(t^{\curlyvee} \to t^{\curlywedge}) + s(t^{\times} \to t^{\curlywedge})\right], \tag{7.3}$$

where $\mathcal{N}(t^{\curlywedge})$ is a randomly sampled set of negative terms such that for any term $t^{\times}$ in the set, $(t^{\times}, t^{\curlywedge}) \notin \mathcal{S}$. For each pair in $\mathcal{S}$, L negative pairs are sampled. By minimizing the above loss, we learn our hypernymy inference model.

7.5 EXPERIMENTS

In this section, we quantitatively evaluate the effectiveness of the proposed method on two real-world large text-rich HINs. An additional case study is also presented by a taxonomy construction downstream applications.

7.5.1 Data Description

Datasets. We use two large-scale real-world HIN datasets. **DBLP** is a bibliographical network in the computer science domain [118], with five node types – *author* (A), *paper* (P), *keyword* (W), *venue* (V), and *year* (Y) – and five edge types – authorship, keyword usage, publishing venue, and publishing year of a paper, and the reference relationship from a paper to another. The text affiliated to a *paper* node is the abstract of the paper, and the text associated with a *keyword* node is the Wikipedia page on this keyword if exists. We let *keyword* be the target node type in the hypernymy discovery task. To generate a set of positive and a set of negative pairs for evaluation, we resort to the ACM Computing Classification System (CCS) [1], which organizes computer science topics into multiple tree structures. A keyword in the vocabulary $\mathcal{D}$ is mapped to a term in CCS if they can be linked to the same Wikipedia entry using a publicly-available tool named WikiLinker [2], a positive hypernym-hyponym label is recorded if two keywords are mapped to two CCS terms that have ancestor-descendant relation in a CCS tree. For each of the $10,055$ positive hypernym-hyponym pairs attained as such, we first generate five negative pairs by fixing the hypernym and randomly sampling a keyword that is not its hyponym as a negative hyponym and then generates another five negative pairs in the same way by fixing the hyponym. In

[1] https://www.acm.org/publications/class-2012

[2] https://github.com/franticnerd/taxogen/blob/dev_jmshen/code/preprocess/wikiLinker.py

Table 7.1: Basic statistics for the DBLP and PSN dataset, where K's stands for thousands and M's represents millions. The number of sentences is reported for the corpus size $|\Gamma|$.

Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{T}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{D}\rvert$	$\lvert\Gamma\rvert$
DBLP	3,715,234	20,594,906	5	5	32,688	10,147,503
PSN	M's	hundreds of M's	5	5	5,000	tens of M's

this negative pair generation process, a keyword is always randomly sampled from the set that can be mapped to CCS terms. **PSN** is an internal profession social network (PSN) data and we anonymize its origin for the double-blind review process. The network has five node types – *user*, *skill*, *employer*, *school*, and *position* – and five edge types, which represent users possessing skills, working for employers, attending schools, holding job positions, and being connected with other users. The text associated to a *skill*, a position, and an employer are respectively the Wikipedia page on this skill, users' description for this position, and job posting description created by this employer. The entire network is downsampled to include only users from a major metropolitan area in the United States as well as nodes and edges directly linked to these users. We further filter skills and keep the top $5,000$ regarding the number of users having a skill. We let *skill* be the target node type. Positive hypernym-hyponym labels were curated by the company that owns the data, and the label curation process is independent of our experiments. We randomly generate the negative pairs likewise.

The schemata of these two HINs are depicted in Figure 7.3 on page 93, and we summarize the statistics of the datasets in Table 7.1.

7.5.2 Baselines and Partial Model

We compare with four different baselines relevant to the proposed framework. **Hearst patterns** (**Hearst**) [80] is a classic pattern based method with high precision and compromised recall. It is used to derive weak supervision pairs in our framework. In the evaluation, we assign a common high score for all extracted pairs and assign a common low score for all other pairs. **LAKI** [132] is a method developed for document representation. One of its easy-to-implement components can be used to infer hypernymy by one DIH measure at the *Simplest* context as well as nodewise embedding features for the relevance of the a term pair. Since LAKI uses both DIH feature and nodewise embedding features, it is used to justify the relatively higher complexity the proposed method has. **Poincaré Embedding** (**Poincaré**) [120] is the pioneering algorithm in a line of research on hyperbolic space embedding. These algorithms can reconstruct hypernymy by embedding a taxonomy, which is

a directed acyclic graph (DAG). This line of work is not designed for hypernymy discovery since the input must be a DAG with directed edges representing hypernymy in order to reconstruct more hypernymy. We take it as a baseline because, to the best of our knowledge, this class of algorithms is the only existing ones that are relevant to hypernymy and take graphs or networks as input. **LexNET** [93] is one state-of-the-art algorithm for hypernymy discovery from a corpus. For each input corpus, it leverages both path-based signals and distributional signals, where a path is an advanced way of defining textual pattern. It is an improved version of the authors' earlier algorithm HypeNET [92]. None of these baselines consider context granularity. To understand the impact of modeling context granularity in each dataset, we also experiment with one ablated version of the HDCG model – **HDCG-wo-CG** – which is the proposed model without pairwise features except for those generated under the *Simplest* context.

7.5.3 Evaluation Metrics and Experiment Setups

Evaluation Metrics. We report evaluation results with two precision metrics and four rank-based metrics. For each model, we first find the predicted score for all pairs in the evaluation set obtained in the way described in Section 7.5.1. In case a model does not generate a predicted score for a pair in the evaluation set, we set the predicted score to a default value slightly smaller than the minimum of all scores predicted by this model. In the event of a tie between the predicted scores of two pairs, we use a randomized method to determine which pair would rank higher, so that a model predicting all ties would get the same evaluation result as random guess. For fairness across different runs of evaluation and across different models, we fix the random seed in the evaluation pipeline.

The two precision metrics we used are precision at k (**P@k**) with $k \in \{100, 1000\}$, which is computed the number of positive pairs among the top-ranked k pairs divided by k. The four rank-based metrics are macro mean average reciprocal rank (**MaMARR**), micro mean average reciprocal rank (**MiMARR**), macro mean largest reciprocal rank (**MaMLRR**), and micro mean largest reciprocal rank (**MiMLRR**). We describe their definition as follows. For each hypernym t in the evaluation data, we group all pairs with t being hypernym together. For each pair, its reciprocal rank (RR) is the reciprocal of the rank of this pair within the group. Since there can be multiple positive pairs in a group, the average reciprocal rank for the group is the average over the RR's of all positive pairs, and the largest reciprocal rank is the largest RR among the RR's of all positive pairs. Finally, we compute the macro mean and the micro mean across all groups to get the final four metrics, where the macro mean

Table 7.2: Quantitative evaluation results of hypernymy discovery from the DBLP and the PSN datasets.

Dateset	DBLP						PSN					
Metric	P@100	P@1000	MaMARR	MiMARR	MaMLRR	MiMLRR	P@100	P@1000	MaMARR	MiMARR	MaMLRR	MiMLRR
Hearst [80]	0.550	0.163	0.071	0.032	0.304	0.534	0.680	0.259	0.071	0.066	0.425	0.580
LAKI [132]	0.180	0.191	0.096	0.038	0.382	0.602	0.870	0.491	0.137	0.133	0.508	0.657
Poincaré [120]	0.110	0.088	0.064	0.028	0.277	0.509	0.110	0.114	0.036	0.028	0.212	0.288
LexNET [93]	0.580	0.337	0.121	0.044	0.463	0.542	0.660	0.529	0.129	0.098	0.534	0.605
HDCG-wo-CG	0.790	0.402	0.148	0.061	0.544	0.757	**0.920**	**0.847**	0.410	0.387	0.809	0.859
HDCG	**0.880**	**0.620**	**0.358**	**0.148**	**0.745**	**0.865**	0.860	0.835	**0.447**	**0.414**	**0.842**	**0.890**

assigns uniform weights for all group when calculating the mean and the micro mean assigns weights proportional to the number of positive pairs in each group.

For all six metrics, greater value indicate better performance. We also note that the optimal value the perfect model can achieve for MiMARR and MaMARR may be smaller than 1. This can be explained with an example where a group has three positive pairs, then the highest average reciprocal rank for this group would be $(1/1 + 1/2 + \ldots + 1/6)/6 = 0.408 < 1$.

Experiment Setups. To determine context by explicit network structure, we use *Grp-by-A*, *Grp-by-V*, *Grp-by-W* for DBLP and *Grp-by-P*, *Grp-by-S*, *Grp-by-U* for PSN. For both datasets, we set the dimension of the embedding space $d = 128$. When determining context by clustering, we select two different values for K: 100 and 10000, which yields two contexts *Clus-100* and *Clus-10000*. As a result, for each of the two datasets, we derive $4 \times 6 = 24$ pairwise features. In the hypernymy inference model, we always set negative sampling rate L to 10, dropout rate for $\varphi(\cdot)$ to 0.7, dropout rate for $\psi(\cdot)$ to 0.1.

For each baseline requiring supervision or embedding for pairs as input, we always use the same ones generated in the HDCG framework, and we always tune the hyperparameters of the baselines to the best on the test dataset.

7.5.4 Quantitative Evaluation Results

The main quantitative evaluation results are presented in Table 7.2. Overall, the HDCG–based methods generally outperform all baselines under all metrics in both datasets by large margins with only one exception for P@100 in the PSN dataset, while the full HDCG model clearly outperformed HDCG-wo-CG in DBLP and had a competitive performance with HDCG-wo-CG in PSN.

Notably, the state-of-the-art corpus-based method LexNET mostly excelled among all baselines. However, it still performed significantly inferior to HDCG and the partial model, which directly demonstrated the benefit of introducing network signals in the task of hy-

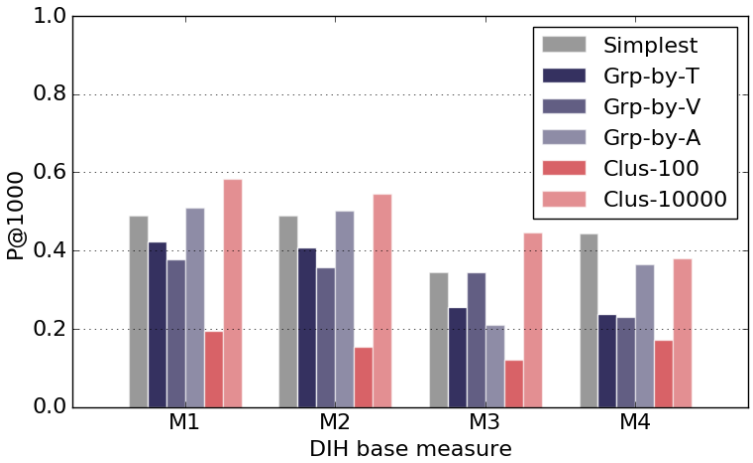


Figure 7.6: The importance of each DIH feature based on different base measures and different context granularities.

pernymy discovery. Also only taking corpus as the input, Hearst further underperformed LexNET under most metrics. It is worth noting that the precision of Hearst dropped drastically when the k in P@k changes from 100 to 1000 in both datasets. In comparison, LexNET had P@100 similar to Hearst, while the former had clearly better P@1000. This outcome further verified the existing observation that Hearst tends to extract a limited number of term pairs, *i.e.*, low recall, while the precision on the extracted pairs could be decent. Poincaré is selected as a baseline because the line of research represented by it is the only one relevant to hypernymy that also takes graphs or networks as input. It had the worst performance among all baselines, which was expected because this algorithm was not designed for discovering hypernymy from data. LAKI also generally performed worse than the two HDCG-based models with one exception for P@100 in PSN. We discuss its relatively better performance in PSN compared to in DBLP in the next paragraph.

The same context granularity can have different importance in different datasets. HDCG clearly outperformed its partial model HDCG-wo-CG in DBLP. This result demonstrated the utility of leveraging multiple context granularities in the DBLP dataset. However, the competition between HDCG and HDCG-wo-CG had mixed results in the PSNdataset. We interpret this as the *Simplest* context is very informative in PSN since each user is linked to more skills on average than each paper is linked to keywords. As a result, introducing more features from other context granularities may not always bring in a significant performance boost. In fact, low-quality noisy features may even dampen the top-ranked pairs resulting in a lower precision especially when k is small. This also explains the better performance of LAKI in PSN, since LAKI did use a DIH measure at the *Simplest* context.

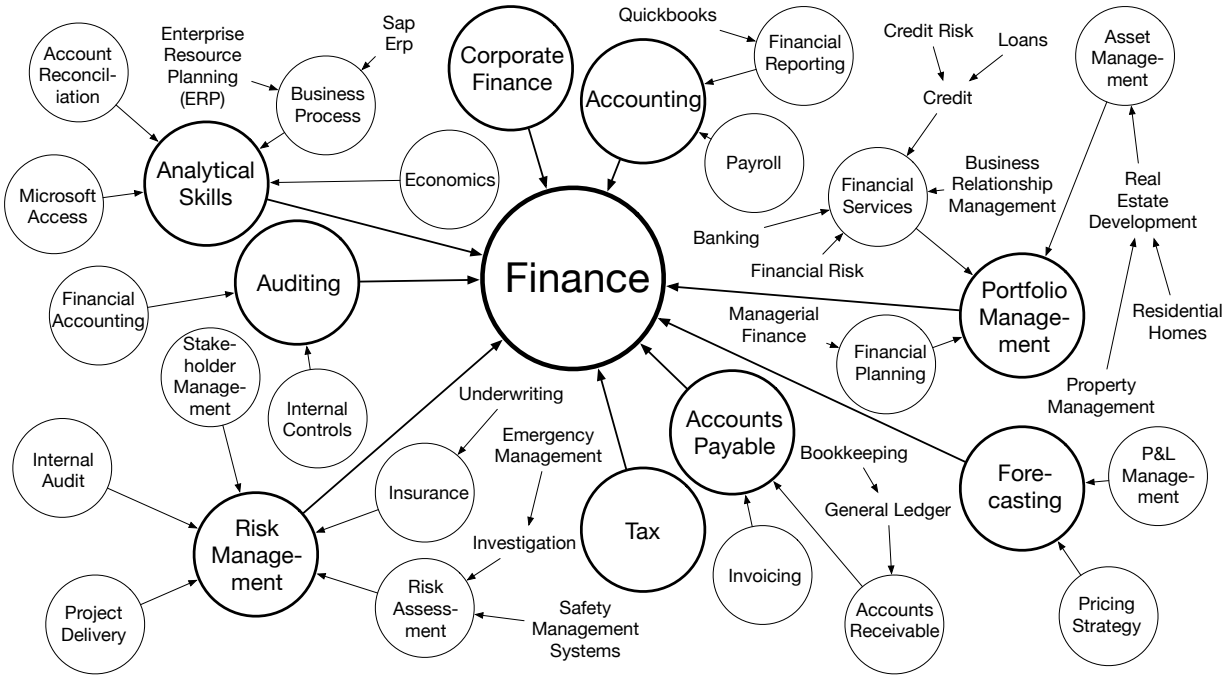


Figure 7.7: Partial view of a skill taxonomy constructed from the hypernymy discovered from the PSN dataset.

Simultaneously leveraging pairwise features from multiple context granularities can introduce performance boost. In Figure 7.6, we plot out the evaluation result using each single network-based DIH feature on the DBLP dataset. We only present the metric P@1000 and omit the other metrics due to space limitations and that similar conclusion can be reached based on other metrics. Comparing Figure 7.6 with Table 7.2, it can be seen that the proposed method using features from multiple context granularities achieved elevated performance compared with every single feature, which corroborates that different hypernymy pairs may be revealed from different context granularity.

No context granularity is always the best even in the same dataset. From the performance of each single feature in Figure 7.6, it can be seen that not a context granularity is the best under all DIH base measures. For instance, *Clus-10000* had the best performance when coupled with M_1, M_2, and M_3, while in the case of M_4, *Simplest* is the best. Also, while *Simplest* is the best with M_4, it is even slightly worse than *Grp-by-A* when coupled with M_1 and M_2.

7.5.5 Case Study: Taxonomy Construction

In this section, we demonstrate the potential of hypernymy discovered by HDCG with the downstream application of taxonomy construction. If we consider each discovered hypernymy pair as a directed edge, putting all pairs together will yield a graph potentially with cycles. However, a taxonomy is always a DAG. We, therefore, resort to a simple heuristic algorithm used in many existing taxonomy construction studies [83, 127], which repeatedly find one cycle in the current graph and then randomly break an edge. We consider the resulting DAG as a crude taxonomy.

Due to the scalability limit of the above cycle breaking algorithm, we construct the initial graph with only 500 most popular skills and then keep only 5000 edges with the highest hypernymy scores.We present the result after the cycle breaking algorithm in Figure 7.7, which include the part of the DAG involving *Finance* and all its hyponyms within the fourth-order neighborhood.

Since only 500 top skills are left when constructing the graph, one should expect the recall of the taxonomy should be limited. The recall aside, the constructed taxonomy has decent overall quality. If we refer to a hyponym of t that also has an edge to t as a child of t, seven out of nine children of *Finance* makes sense except *Tax* and *Analytical Skills*, where the latter two are related to *Finance* but are not precisely its hyponyms. One level deeper, descendants of the seven children are reasonable as well, which further corroborated the effectiveness of the proposed HDCG framework.

As a final remark, even when such unsupervised approach cannot yield a perfect taxonomy, the discovered hypernymy pair with hypernymy scores are still useful when human labelers wish to seek recommendation in growing a taxonomy from roots to leaves.

7.6 SUMMARY

We proposed to discover hypernymy from text-rich HINs, which availed us with additional rich signals from the network data besides corpus. From real-world data, we identified the importance of modeling context granularity in the distributional inclusion hypothesis (DIH). We then proposed the HDCG framework that exploits both network and textual signals for the problem of hypernymy discovery. Experiments and case study demonstrated the effectiveness of the proposed framework as well as the utility of considering context granularity.

CHAPTER 8: DISCUSSION

In this section, we collectively discuss multiple conceptual and practical issues regarding heterogeneous association and its implication.

Sensitivity to noise. Networks derived from real-world scenarios can be inaccurate. Such inaccuracy introduces noise to datasets used as input network mining algorithms. It is hence of interest to understand how the benefit of modeling heterogeneous association, as well as the methods proposed in this dissertation, can be dampened by such noise. Among all the methods proposed in this dissertation, AspEm is the only one whose performance can potentially be not continues to the variation of noise. This is because AspEm involves hard-partitioning the input network to find aspects, which itself is a discretizing process. We, therefore, choose AspEm as an example to perform noise sensitivity study.

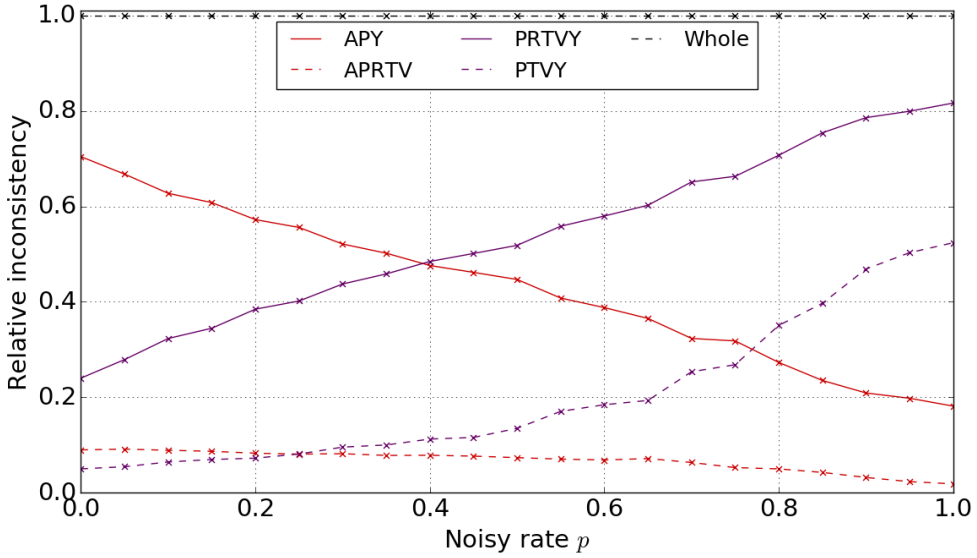


Figure 8.1: Aspect inconsistency under varied noisy rate normalized by the inconsistency of the whole network.

To this end, we again use the DBLP dataset and the link prediction problem introduced in Section 5.4 as the downstream evaluation task. To introduce noise to the input dataset, with probability p, we replace each edge with another edge connecting the original center node to a randomly selected node with type identical to that of the original node linked to the center node. In other words, the expected noise rate of the new dataset is p.

In Figure 8.1, we plot out the relative inconsistency score of multiple aspects of interests against the noise rate, where p traverses 0.00, 0.05, 0.10, . . . , 1.00. The relative inconsistency

Table 8.1: Link prediction results on DBLP on noisy network data under varied noise rate.

Method	OneSpace						AspEm					
Metrics	$P@1$	$P@3$	$P@10$	$R@1$	$R@3$	$R@10$	$P@1$	$P@3$	$P@10$	$R@1$	$R@3$	$R@10$
0.0	0.7440	0.5381	0.2279	0.3301	0.6401	0.8519	0.7724	0.5645	0.2356	0.3479	0.6749	0.8810
0.1	0.7423	0.5319	0.2225	0.3280	0.6303	0.8292	0.7635	0.5531	0.2302	0.3424	0.6587	0.8596
0.2	0.7327	0.5193	0.2167	0.3221	0.6139	0.8085	0.7425	0.5319	0.2227	0.3297	0.6317	0.8317
0.4	0.6837	0.4737	0.2003	0.2941	0.5532	0.7448	0.6912	0.4860	0.2049	0.3001	0.5712	0.7612
0.6	0.5883	0.3954	0.1721	0.2440	0.4561	0.6388	0.5973	0.4074	0.1757	0.2496	0.4709	0.6499
0.8	0.3826	0.2516	0.1192	0.1480	0.2815	0.4380	0.3819	0.2508	0.1181	0.1486	0.2796	0.4342
1.0	0.0272	0.0278	0.0274	0.0107	0.0311	0.1024	0.0274	0.0275	0.0273	0.0103	0.0303	0.1017

of an aspect is the inconsistency of this aspect normalized by that of the whole network. Recall that the two representative aspects selected are {APRTV, APY}. The rank of inconsistency score of APRTV or APY first changed when the score of APRTV surpasses the score of PTVY at noise rate between 0.20 and 0.25. In other words, since the unchanged rank guarantees unaltered aspect selection, the aspect selection result is robust to noise with noise rate at least up to 0.20.

Furthermore, we study how noise impacts the quantitative evaluation result of the link prediction task in DBLP. For each noisy input data generated as described above, we learn the embedding with OneSpace and the proposed method with representative aspects {APRTV, APY}. OneSpace is used for comparison because (i) it serves as a direct comparison to the proposed method and can be used to verify the utility of resolving incompatibility by aspects and (ii) it is among the best baselines in previous experiments. It is worth noting that the training data in the downstream learner is not intruded by noises so that we can focus on the impact of noise on the embedding algorithm itself. This is experiment, we set p to be 0.0, 0.1, 0.2, 0.4, 0.6, 0.8, or 1.00. The results are presented in Table 8.1.

Overall, the performance of each model continuously drops as the noise rate increases. When the noise rate is 1, the input HIN has only random edges, and both methods degenerate to random guess. Notably, the relative difference between OneSpace and AspEm decreased to half of that without noise only after the noise rate increases to 0.2. This implies AspEm could still be a valid method with the presence of noise.

Possibility of general-purpose representation learning that subsumes the proposed methods. In this dissertation, we demonstrated the existence of heterogeneous association in real-world networks and introduced multiple embedding learning methods to explicitly model such heterogeneous association. On the other hand, it would be appealing if researchers could devise a general-purpose representation learning that subsumes the proposed methods so that without explicitly modeling heterogeneous association by means in

AspEm or HEER, one can still harness such heterogeneity in data. We argue that while such direction is very attempting, it is also a difficult problem, and many challenges are still unsolved before the research community can approach its solution. We also note that a balance of information loss and compactness exists in pursuing such solution – the most comprehensive representation is the sparse representation of adjacency matrix of the network itself, which is not compact or directly useful and is simultaneously lossless. As such, we consider the proposed methods to be one step toward the general-purpose representation learning algorithm in the sense that the latter should have the parameter space to embed the heterogeneous association.

Possibility of automatic data interpretation. As we have observed the existence of heterogeneous association in real-world networks, it is of interest to understand whether it is possible to better automatically interpret data considering such heterogeneity. Without supervision, we have proposed an incompatibility measure in Section 5.3.1 to quantify the incompatibility within a network. For the specific task of deriving relevance measure, Chapter 4 described a relevance measure that can reveal the heterogeneous association among meta-paths. We note that for general purpose, the incompatibility measure discussed in Section 5.3.1 is one way to characterize such heterogeneity, but this measure is not the only reasonable measure, and different measures can potentially prefer different tasks. Therefore, such unsupervised measures can be a handy approach to obtain a quick insight from the dataset, while in case the concerned task is specified, it could be beneficiary to interpret data with a method considering the objective of the task. In this way, the heterogeneity identified could be more pertinent to the task of interest.

Scenario where the proposed methods would not work. The methods proposed in the dissertation are all based on the necessity of modeling heterogeneous association. As a result, when such heterogeneity does not exist in the network, the benefit of modeling it would also vanish. An extreme example is when incompatibility does not exist among network component, AspEm would not be able to identify semantically distinct representative aspects. In this case, crafting multiple network partitions as aspects would not possibly introduce performance boost compared with using the whole network. Similarly, if such heterogeneity does not exist, HEER would not benefit from employing heterogeneous metrics either.

Further clustering aspect in modeling heterogeneous association. The incompatibility measure proposed in Section 5.3.1 provides an unsupervised approach to quantify incompatibility among network components. Using such an approach one can further measure the closeness among aspects. As such, it is possible to further cluster aspects to construct an

aspect hierarchy. The embedding algorithm can potentially be further improved with such aspect hierarchy for modeling multiple levels of incompatibility within the network. The method for embedding leveraging aspect hierarchy is yet to be studied, while a few straightforward methods exist. One example is to first learn the embedding using higher-level, more coarse aspects as pre-training, and then proceed to lower-level, more specific aspects for fine-tuning.

The impact of bias in hypernymy discovery by HDCG. For both *keyword* in DBLP and *skill* in PSN, the input dataset can have a bias toward the popular target nodes. Specifically, these target nodes would have higher degrees in the HIN and more occurrences in the associated corpus. We note that such bias in the corpus could lead to lexical memorization in our model. Meanwhile, its impact from the HIN is less prominent since the generality of hypernymy pairs is coupled with the granularity of contexts. The less popular nodes – which is semantically less general – would be addressed appropriately at a finer granularity.

Factors impacting the performance of context in HDCG. As we have demonstrated through illustrative examples and experiments, the granularity is a critical factor of a context that impacts the discovery ability of hypernymy measures under this context. Meanwhile, it is worth noting that granularity is not the sole factor, and the more specific semantics of the context is useful in further explaining the behavior of different contexts. As an example, the finer *Clus-10000* context has better performance than the coarser *Clus-100* under each measure according to Figure 7.6. However, specifically under **M3**, the coarser *Grp-by-V* outperformed the finer *Grp-by-A*. A more extreme example is defining each context unit to be a randomly generated cluster. In this case, any context will always give random results regardless of granularity.

Confidence of the induced taxonomy by HDCG. In the application of taxonomy or ontology, it is often useful to have an estimate on the confidence of each edge. In our case study in Section 7.5.5, we did not provide confidence for the constructed taxonomy. We note that generating such a confidence score is possible since the hypernymy inference model output hypernymy pairs with likelihood, and several DAG construction algorithms can factor in edge weights. However, such estimate on confidence is prone to error propagation in the DAG construction process. As such, if one wishes to improve the confidence estimate for such two-step taxonomy construction method, it would be interesting to propose a DAG construction method that specifically serves this purpose. Furthermore, it is worth noting that the end-to-end method for taxonomy construction exists in the literature that circumvents the error propagation problem in the DAG construction process [85].

CHAPTER 9: CONCLUSION

In this dissertation, I have studies the heterogeneous association across different components of typed real-world networks. Previously, the studies of typed networks mainly leverage the benefit introduced by the distinct node or edge types per se, while this dissertation focuses on the higher-level heterogeneous association, which we identify as important and intrinsic to the signals encapsulated in the networks. The proposed principles and techniques can help the researchers to investigate networks from a new perspective and enable the developers to further unleash the power of the ubiquitous typed real-world network data. The major contributions of this work are summarized as follows.

- **Identified the importance of heterogeneous association in real-world networks.** When real-world networked data come with type information, the network would often consist of multiple distinct components that are backed by the different generating mechanism. As a result, assuming the homogeneous association among the components is a stronger assumption in the typed real-world networks than that it is in the untyped, homogeneous networks. Through a series of analyses of different real-world datasets in multiple data mining tasks, I showcased the benefit of modeling heterogeneous association.

- **Developed easy-to-use tools to harness heterogeneous association.** Aside from the multiple models proposed for different data mining tasks in Chapter 3, 4, and 7, I provide a solution for users who wish to quickly enjoy the benefit of modeling heterogeneous without tweaking or customizing their own model. The users can embed their typed network using one algorithm I provide, and then use any appropriate off-the-shelf learning algorithm on top of the learned embedding to complete their task. In this dissertation, two algorithms considering heterogeneous association are developed: AspEm in Chapter 5 and HEER in Chapter 6. AspEm would be recommended in the case where the users have the knowledge that multiple aspects with distinctively incompatible semantics exist in the network, or the users have the expertise to assign meaningful semantic aspects. On the other hand, if the users have little knowledge of the network or many (*e.g.*, more than ten) edge types exist in the network, I would recommend first try out HEER as the embedding algorithm.

- **Demonstrated the viability of practicing the proposed principle with little customization effort to applications.** In Chapter 7, I tackled the problem of hypernymy discovery with network data, which is outside of the scope of previous

network mining studies to the best of my knowledge. I demonstrated the viability of applying the principle of modeling heterogeneous association within typed real-world networks to this problem – an important and fundamental problem in natural language processing. In the proposed solution, the framework design for modeling heterogeneous association is relatively simple and straightforward, which showed it is possible to apply the proposed principle to application scenarios with little effort in designing or tweaking models customized for the application.

While working on this dissertation, I find several promising directions to further the frontiers of studying heterogeneous association in real-world networks. I discuss two of them in as follows.

The first one is to study task-specific approaches to model the heterogeneous association. While I have observed that the association strength among network components can vary, it is possible only a portion of the components, and their associations are relevant to given downstream tasks. However, this dissertation either models heterogeneous association in an unsupervised fashion or only inject supervision in the second stage of a two-stage solution where the heterogeneous association is considered in the first stage. For example, the AspEm algorithm select aspects in an unsupervised way with an incompatibility measure. In case the downstream task is given, or certain supervision is available, it would be instrumental in selecting more aspects more pertinent to the task especially when the space for candidate aspects is large.

The other direction is to understand and model the impact of heterogeneous association in data standardization. Data standardization is a widely used process to convert data from multiple sources to a standardized format so that users can access the data more easily in a standardized approach. In the scenario of network data, examples of data standardization include managing multiple typed social networks together and constructing biological networks from multiple sources. The networks involved in these scenarios are typed in nature, and the heterogeneous association may prevalently exist. Meanwhile, the outcome of data standardization is expected to have a common format and be accessible in a standardized approach. Therefore, it is of interests to understand the impact of the existence of heterogeneous association on the outcome of standardizing data, and in case it negatively affects data standardization, what are the principles and methodologies to alleviate this problem. The study in this direction can potentially benefit collective research in all data-driven fields as well as the companies in the industry that generate and profit from large-scale heterogeneous data.

REFERENCES

[1] Y. Sun, J. Han, X. Yan, and P. S. Yu, "Mining knowledge from interconnected data: a heterogeneous information network analysis approach," in *VLDB*, 2012.

[2] C. Shi, Y. Li, J. Zhang, Y. Sun, and S. Y. Philip, "A survey of heterogeneous information network analysis," *TKDE*, vol. 29, no. 1, pp. 17–37, 2017.

[3] A. Kumar and H. Daumé, "A co-training approach for multi-view spectral clustering," in *ICML*, 2011, pp. 393–400.

[4] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," in *NIPS*, 2011, pp. 1413–1421.

[5] J. Liu, C. Wang, J. Gao, and J. Han, "Multi-view clustering via joint nonnegative matrix factorization," in *SDM*, vol. 13. SIAM, 2013, pp. 252–260.

[6] D. Zhou and C. J. Burges, "Spectral clustering and transductive learning with multiple views," in *ICML*. ACM, 2007, pp. 1159–1166.

[7] V. Sindhwani and P. Niyogi, "A co-regularized approach to semi-supervised learning with multiple views," in *ICML Workshop on Learning with Multiple Views*, 2005.

[8] D. Zhang, F. Wang, C. Zhang, and T. Li, "Multi-view local learning." in *AAAI*, 2008, pp. 752–757.

[9] H. Hu, X. Yan, Y. Huang, J. Han, and X. J. Zhou, "Mining coherent dense subgraphs across massive biological networks for functional discovery," *Bioinformatics*, vol. 21, no. suppl 1, pp. i213–i221, 2005.

[10] J. Pei, D. Jiang, and A. Zhang, "On mining cross-graph quasi-cliques," in *KDD*. ACM, 2005, pp. 228–238.

[11] Z. Zeng, J. Wang, L. Zhou, and G. Karypis, "Coherent closed quasi-clique discovery from large dense graph databases," in *KDD*. ACM, 2006, pp. 797–802.

[12] O. Frank and K. Nowicki, "Exploratory statistical anlaysis of networks," *Annals of Discrete Mathematics*, vol. 55, pp. 349–365, 1993.

[13] P. Pattison and S. Wasserman, "Logit models and logistic regressions for social networks: II. multivariate relations," *British Journal of Mathematical and Statistical Psychology*, vol. 52, no. 2, pp. 169–194, 1999.

[14] Y. Sun and J. Han, "Mining heterogeneous information networks: a structural analysis approach," *SIGKDD Explorations*, vol. 14, no. 2, pp. 20–28, 2013.

[15] Y. Sun, Y. Yu, and J. Han, "Ranking-based clustering of heterogeneous information networks with star network schema," in *KDD*, 2009.

[16] H. Zhuang, J. Zhang, G. Brova, J. Tang, H. Cam, X. Yan, and J. Han, "Mining query-based subnetwork outliers in heterogeneous information networks," in *ICDM*, 2014.

[17] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *WSDM*. ACM, 2014.

[18] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*, 2016.

[19] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *KDD*, 2014.

[20] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *WWW*, 2015.

[21] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *KDD*, 2016.

[22] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," in *VLDB*, 2011.

[23] N. Lao and W. W. Cohen, "Relational retrieval using a combination of path-constrained random walks," *Machine learning*, vol. 81, no. 1, pp. 53–67, 2010.

[24] J. He, J. Bailey, and R. Zhang, "Exploiting transitive similarity and temporal dynamics for similarity search in heterogeneous information networks," in *DASFAA*. Springer, 2014.

[25] K. Yao, H. F. Mak et al., "Pathsimext: Revisiting pathsim in heterogeneous information networks," in *WAIM*. Springer, 2014.

[26] Y. Fang, W. Lin, V. W. Zheng, M. Wu, K. C.-C. Chang, and X.-L. Li, "Semantic proximity search on graphs with metagraph-based learning," in *ICDE*. IEEE, 2016.

[27] Z. Huang, Y. Zheng, R. Cheng, Y. Sun, N. Mamoulis, and X. Li, "Meta structure: Computing relevance in large heterogeneous information networks," in *KDD*. ACM, 2016.

[28] C. Shi, X. Kong, Y. Huang, S. Y. Philip, and B. Wu, "Hetesim: A general framework for relevance measure in heterogeneous networks," *TKDE*, vol. 26, no. 10, pp. 2479–2492, 2014.

[29] T. Chen and Y. Sun, "Task-guided and path-augmented heterogeneous network embedding for author identification," in *WSDM*, 2017.

[30] C. Wang, R. Raina, D. Fong, D. Zhou, J. Han, and G. Badros, "Learning relevance from heterogeneous social network and its application in online targeting," in *SIGIR*. ACM, 2011.

[31] X. Yu, Y. Sun, B. Norick, T. Mao, and J. Han, "User guided entity similarity search using meta-path selection in heterogeneous information networks," in *CIKM*. ACM, 2012.

[32] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *KDD*, 2015.

[33] H. Gui, J. Liu, F. Tao, M. Jiang, B. Norick, and J. Han, "Large-scale embedding learning in heterogeneous event data," in *ICDM*, 2016.

[34] J. Tang, M. Qu, and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks," in *KDD*. ACM, 2015.

[35] F. Battiston, V. Nicosia, and V. Latora, "Structural measures for multiplex networks," *Physical Review E*, 2014.

[36] M. De Domenico, A. Solé-Ribalta, E. Cozzo, M. Kivelä, Y. Moreno, M. A. Porter, S. Gómez, and A. Arenas, "Mathematical formulation of multilayer networks," *Physical Review X*, 2013.

[37] M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, and D. Pedreschi, "Multidimensional networks: foundations of structural analysis," *World Wide Web*, 2013.

[38] Y. Shi, P.-W. Chan, H. Zhuang, H. Gui, and J. Han, "Prep: Path-based relevance from a probabilistic perspective in heterogeneous information networks," in *KDD*, 2017.

[39] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *KDD*, 2016, pp. 1105–1114.

[40] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "struc2vec: Learning node representations from structural identity," in *KDD*, 2017.

[41] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv preprint arXiv:1709.05584*, 2017.

[42] H. Dai, B. Dai, and L. Song, "Discriminative embeddings of latent variable models for structured data," in *ICML*, 2016.

[43] M. Zhang and Y. Chen, "Weisfeiler-lehman neural machine for link prediction," in *KDD*, pp. 575–583.

[44] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.

[45] L. Xu, X. Wei, J. Cao, and P. S. Yu, "On exploring semantic meanings of links for embedding social networks," in *WWW*, 2018.

[46] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *ICML*, 2016.

[47] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *ICLR*, 2018.

[48] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[49] S. Abu-El-Haija, B. Perozzi, and R. Al-Rfou, "Learning edge representations via low-rank asymmetric projections," in *CIKM*, 2017.

[50] M. Qu, J. Tang, J. Shang, X. Ren, M. Zhang, and J. Han, "An attention-based collaboration framework for multi-view network representation learning," in *CIKM*. ACM, 2017.

[51] J. Ni, S. Chang, X. Liu, W. Cheng, H. Chen, D. Xu, and X. Zhang, "Co-regularized deep multi-network embedding," in *WWW*, 2018.

[52] I. Gollini and T. B. Murphy, "Joint modelling of multiple network views," *Journal of Computational and Graphical Statistics*, pp. 00–00, 2014.

[53] D. Greene and P. Cunningham, "Producing a unified graph representation from multiple social network views," in *Web Science Conference*. ACM, 2013, pp. 118–121.

[54] T.-y. Fu, W.-C. Lee, and Z. Lei, "Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning," in *CIKM*, 2017.

[55] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *KDD*, 2017.

[56] J. Shang, M. Qu, J. Liu, L. M. Kaplan, J. Han, and J. Peng, "Meta-path guided embedding for similarity search in large-scale heterogeneous information networks," *arXiv preprint arXiv:1610.09769*, 2016.

[57] Y. Shi, H. Gui, Q. Zhu, L. Kaplan, and J. Han, "Aspem: Embedding learning by aspects in heterogeneous information networks." in *SDM*, 2018.

[58] Z. Liu, V. W. Zheng, Z. Zhao, F. Zhu, K. C.-C. Chang, M. Wu, and J. Ying, "Semantic proximity search on heterogeneous graph by proximity embedding," in *AAAI*, 2017.

[59] S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang, "Tri-party deep network representation," in *IJCAI*, 2016.

[60] S. Arora, Y. Li, Y. Liang, T. Ma, and A. Risteski, "Linear algebraic structure of word senses, with applications to polysemy," *arXiv preprint arXiv:1601.03764*, 2016.

[61] S. K. Jauhar, C. Dyer, and E. H. Hovy, "Ontologically grounded multi-sense representation learning for semantic vector space models." in *HLT-NAACL*, 2015, pp. 683–693.

[62] A. Neelakantan, J. Shankar, A. Passos, and A. McCallum, "Efficient non-parametric estimation of multiple embeddings per word in vector space," *arXiv preprint arXiv:1504.06654*, 2015.

[63] S. Šuster, I. Titov, and G. van Noord, "Bilingual learning of multi-sense embeddings with discrete autoencoders," *arXiv preprint arXiv:1603.09128*, 2016.

[64] G. Jeh and J. Widom, "Simrank: a measure of structural-context similarity," in *KDD*. ACM, 2002.

[65] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[66] J. Kuck, H. Zhuang, X. Yan, H. Cam, and J. Han, "Query-based outlier detection in heterogeneous information networks," in *ICDE*, 2015.

[67] C. Wang, X. He, and A. Zhou, "A short survey on taxonomy learning from text corpora: Issues, resources and recent advances," in *EMNLP*, 2017.

[68] P. D. Turney and P. Pantel, "From frequency to meaning: Vector space models of semantics," *Journal of artificial intelligence research*, vol. 37, pp. 141–188, 2010.

[69] D. Lin et al., "An information-theoretic definition of similarity." in *Icml*, vol. 98, no. 1998. Citeseer, 1998, pp. 296–304.

[70] M. Geffet and I. Dagan, "The distributional inclusion hypotheses and lexical entailment," in *ACL*, 2005.

[71] M. Zhitomirsky-Geffet and I. Dagan, "Bootstrapping distributional feature vector quality," *Computational linguistics*, vol. 35, no. 3, pp. 435–461, 2009.

[72] J. Weeds, D. Weir, and D. McCarthy, "Characterising measures of lexical distributional similarity," in *COLING*, 2004.

[73] L. Kotlerman, I. Dagan, I. Szpektor, and M. Zhitomirsky-Geffet, "Directional distributional similarity for lexical inference," *Natural Language Engineering*, vol. 16, no. 4, pp. 359–389, 2010.

[74] D. Clarke, "Context-theoretic semantics for natural language: an overview," in *ACL-GEMS*, 2009.

[75] A. Lenci and G. Benotto, "Identifying hypernyms in distributional semantic spaces," in *ACL-SEM*, 2012.

[76] M. Rei and T. Briscoe, "Looking for hyponyms in vector space," in *CoNLL*, 2014.

[77] E. Santus, A. Lenci, Q. Lu, and S. Schulte im Walde, "Chasing hypernyms in vector spaces with entropy," in *EACL*, 2014.

[78] L. Rimell, "Distributional lexical entailment by topic coherence," in *EACL*, 2014.

[79] S. Roller, K. Erk, and G. Boleda, "Inclusive yet selective: Supervised distributional hypernymy detection," in *COLING*, 2014.

[80] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *Proceedings of the 14th conference on Computational linguistics-Volume 2*. Association for Computational Linguistics, 1992, pp. 539–545.

[81] W. Wu, H. Li, H. Wang, and K. Q. Zhu, "Probase: A probabilistic taxonomy for text understanding," in *SIGMOD*, 2012.

[82] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, "Web-scale information extraction in knowitall:(preliminary results)," in *WWW*, 2004.

[83] Z. Kozareva and E. Hovy, "A semi-supervised method to learn and construct taxonomies using the web," in *EMNLP*, 2010.

[84] Z. Kozareva, E. Riloff, and E. Hovy, "Semantic class learning from the web with hyponym pattern linkage graphs," *Proceedings of ACL-08: HLT*, pp. 1048–1056, 2008.

[85] Y. Mao, X. Ren, J. Shen, X. Gu, and J. Han, "End-to-end reinforcement learning for automatic taxonomy induction," *arXiv preprint arXiv:1805.04044*, 2018.

[86] A. Ritter, S. Soderland, and O. Etzioni, "What is this, anyway: Automatic hypernym discovery." in *AAAI Spring Symposium: Learning by Reading and Learning to Read*, 2009, pp. 88–93.

[87] T. L. Anh, J.-j. Kim, and S. K. Ng, "Taxonomy construction using syntactic contextual evidence," in *EMNLP*, 2014.

[88] R. Snow, D. Jurafsky, and A. Y. Ng, "Learning syntactic patterns for automatic hypernym discovery," in *Advances in neural information processing systems*, 2005, pp. 1297–1304.

[89] N. Nakashole, G. Weikum, and F. Suchanek, "Patty: a taxonomy of relational patterns with semantic types," in *EMNLP*, 2012.

[90] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: A web-scale approach to probabilistic knowledge fusion," in *KDD*, 2014.

[91] R. Snow, D. Jurafsky, and A. Y. Ng, "Semantic taxonomy induction from heterogenous evidence," in *ACL*, 2006.

[92] V. Shwartz, Y. Goldberg, and I. Dagan, "Improving hypernymy detection with an integrated path-based and distributional method," *arXiv preprint arXiv:1603.06076*, 2016.

[93] V. Shwartz and I. Dagan, "Path-based vs. distributional information in recognizing lexical semantic relations," *arXiv preprint arXiv:1608.05014*, 2016.

[94] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu, "Learning semantic hierarchies via word embeddings," in *ACL*, 2014.

[95] T. L. Anh, Y. Tay, S. C. Hui, and S. K. Ng, "Learning term embeddings for taxonomic relation identification using dynamic weighting neural network," in *EMNLP*, 2016.

[96] Z. Yu, H. Wang, X. Lin, and M. Wang, "Learning term embeddings for hypernymy identification." in *IJCAI*, 2015.

[97] O. Levy, S. Remus, C. Biemann, and I. Dagan, "Do supervised distributional methods really learn lexical inference relations?" in *NAACL*, 2015.

[98] M. Rei, D. Gerz, and I. Vulić, "Scoring lexical entailment with a supervised directional similarity network," *arXiv preprint arXiv:1805.09355*, 2018.

[99] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[100] A. Mnih and K. Kavukcuoglu, "Learning word embeddings efficiently with noise-contrastive estimation," in *Advances in neural information processing systems*, 2013, pp. 2265–2273.

[101] I. Iacobacci, M. T. Pilehvar, and R. Navigli, "Sensembed: Learning sense embeddings for word and relational similarity," in *ACL*, 2015.

[102] B. Perozzi, V. Kulkarni, H. Chen, and S. Skiena, "Don't walk, skip! online learning of multi-scale network embeddings," in *ASONAM*, 2017.

[103] J. P. Alston, "Wa, guanxi, and inhwa: Managerial principles in japan, china, and korea," *Business Horizons*, vol. 32, no. 2, pp. 26–31, 1989.

[104] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *NIPS*, 2011.

[105] R. Zafarani and H. Liu, "Social computing data repository at ASU," 2009. [Online]. Available: http://socialcomputing.asu.edu

[106] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, June 2014.

[107] Y. Xiong, Y. Zhu, and S. Y. Philip, "Top-k similarity join in heterogeneous information networks," *TKDE*, vol. 27, no. 6, pp. 1710–1723, 2015.

[108] Y. Shi, M. Kim, S. Chatterjee, M. Tiwari, S. Ghosh, and R. Rosales, "Dynamics of large multi-view social networks: Synergy, cannibalization and cross-view interplay," in *KDD*. ACM, 2016.

[109] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[110] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013, vol. 87.

[111] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the l 1-ball for learning in high dimensions," in *ICML*. ACM, 2008.

[112] J. J. McAuley and J. Leskovec, "Learning to discover social circles in ego networks." in *NIPS*, vol. 2012, 2012, pp. 548–56.

[113] J. Tang, A. C. Fong, B. Wang, and J. Zhang, "A unified probabilistic framework for name disambiguation in digital library," *TKDE*, vol. 24, no. 6, pp. 975–987, 2012.

[114] R. A. Horn, "The hadamard product," in *Proc. Symp. Appl. Math*, vol. 40, 1990, pp. 87–169.

[115] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numerische mathematik*, vol. 14, no. 5, pp. 403–420, 1970.

[116] B. Zhang and M. A. Hasan, "Name disambiguation in anonymized graphs using network embedding," in *CIKM*, 2017.

[117] C. Zhang, L. Liu, D. Lei, Q. Yuan, H. Zhuang, T. Hanratty, and J. Han, "Triovecevent: Embedding-based online local event detection in geo-tagged tweet streams," in *KDD*, 2017.

[118] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: extraction and mining of academic social networks," in *KDD*, 2008.

[119] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *WWW*, 2007.

[120] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *NIPS*, 2017.

[121] H. Deng, J. Han, B. Zhao, Y. Yu, and C. X. Lin, "Probabilistic topic models with biased propagation on heterogeneous information networks," in *KDD*, 2011.

[122] H. Deng, B. Zhao, and J. Han, "Collective topic modeling for heterogeneous networks," in *SIGIR*, 2011.

[123] C. Wang, Y. Song, A. El-Kishky, D. Roth, M. Zhang, and J. Han, "Incorporating world knowledge to document clustering via heterogeneous information networks," in *KDD*, 2015.

[124] C. Wang, Y. Song, H. Li, Y. Sun, M. Zhang, and J. Han, "Distant meta-path similarities for text-based heterogeneous information networks," in *CIKM*, 2017.

[125] J. Yamane, T. Takatani, H. Yamada, M. Miwa, and Y. Sasaki, "Distributional hypernym generation by jointly learning clusters and projections," in *COLING*, 2016.

[126] I. Vulić, D. Gerz, D. Kiela, F. Hill, and A. Korhonen, "Hyperlex: A large-scale evaluation of graded lexical entailment," *Computational Linguistics*, vol. 43, no. 4, pp. 781–835, 2017.

[127] P. Velardi, S. Faralli, and R. Navigli, "Ontolearn reloaded: A graph-based algorithm for taxonomy induction," *Computational Linguistics*, vol. 39, no. 3, pp. 665–707, 2013.

[128] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *TKDE*, 2018.

[129] Y. Shi, Q. Zhu, F. Guo, C. Zhang, and J. Han, "Easing embedding learning by comprehensive transcription of heterogeneous information networks," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 2190–2199.

[130] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a siamese time delay neural network," in *IJPRAI*, 1993.

[131] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *CVPR*, 2006.

[132] J. Liu, X. Ren, J. Shang, T. Cassidy, C. R. Voss, and J. Han, "Representing documents via latent keyphrase inference," in *WWW*. International World Wide Web Conferences Steering Committee, 2016, pp. 1057–1067.